Capitolo 6

Applicazione pratica

> *"In architecture, the vertical integration of computer-based design and manufacturing is giving rise to new forms of digital artisanship, narrowing the albertian divide between conceivers and makers. Likewise, the digitally enhanced horizontal integration of actors and agencies in the design and production process is already chellenging the modern notion of the architect's full authotial control and intellectual ownership of the end product."*[1]
>
> (Carpo 2011, p. 117)

Al fine di sperimentare il funzionamento degli strumenti pratici definiti nel capitolo precedente e di verificarne la correttezza, si è deciso di attivare un'iniziativa di Architettura Open Source presso il Politecnico di Torino, iniziativa cha ha preso il nome di BrickShell. Come verrà meglio spiegato in seguito, l'esperienza di BrickShell ha avuto luogo all'interno di un workshop opzionale proposto agli studenti delle lauree magistrali in Architettura. BrickShell non è stata l'unica sperimentazione realizzata, ma è la più completa e la più utile, dal momento che è stata l'occasione per applicare la quasi totalità degli strumenti pratici messi a punto durante il lavoro di ricerca.

In seguito si riassumono brevemente alcune esperienze personali relative all'Architettura Open Source e alla gestione di piattaforme online. Nel giugno 2011, poco dopo aver iniziato questo lavoro di tesi, vi è stata la partecipazione all'iniziativa promossa da OpenSimSim focalizzata sull'emergenza

[1] "In architettura, l'integrazione verticale tra progettazione digitale e produzione sta dando adito a nuove forme di artigianato digitale, ricucendo la spaccatura albertiana tra ideatori e fabbricanti. Allo stesso modo, l'integrazione orizzontale tra i diversi soggetti coinvolti nel processo progettuale e costruttivo sta minando la nozione moderna di controllo autoriale da parte dell'architetto e di proprietà intellettuale del prodotto finale." [traduzione italiana a cura dell'autore].

post-sisma nipponico. L'esperienza di OpenJapan, già illustrata in precedenza, nonostante gli evidenti limiti organizzativi e strutturali, ha avuto il merito di suscitare questioni e dubbi da cui sono scaturite molte delle riflessioni scritte fin qui. Il secondo corpus di esperienze significative è stato il lavoro di creazione di piattaforme di sostegno alla didattica per due atelier di progettazione architettonica presso il Politecnico di Torino: l'atelier del secondo anno della laurea triennale intitolato *Lo spazio dell'abitare*, tenuto dai professori Pierr-Alain Croset ed Edoardo Piccoli, e gli atelier del primo anno della laurea magistrale in *Architettura, Costruzione e Città*, in cui si è lavorato sulla città di Kigali (nel 2012) e sulla città di Cusco (2013) tenuti dai professori Pierre-Alain Croset, Angelo Sampieri e Simonetta Pagliolico. In queste occasioni è stata offerta la possibilità di progettare e realizzare piattaforme di condivisione pensate appositamente come supporto alla didattica[2]. Si è quindi potuto assistere in prima persona alla formazione di comunità di utenti focalizzate sull'architettura, e alle dinamiche che si innescano grazie a uno strumento collaborativo tra i vari partecipanti.

Seppur non ancora focalizzate sull'Architettura Open Source così come è stata definita nei capitoli precedenti, le esperienze pregresse sono state comunque molto utili per comprendere appieno il significato della cosiddetta 'rivoluzione digitale' e cominciare a ragionare sulle possibili implicazioni che questa potrebbe avere sulle relazioni tra formazione dell'architetto, metodologia progettuale ed evoluzione del processo edilizio.

6.1 BrickShell

Nel gennaio 2013 è iniziata la pianificazione didattica del workshop opzionale *Morfogenesi Computazionale*, tenuto dal professore Mario Sassone. Il corso si rivolgeva a tutti gli studenti delle lauree magistrali in architettura del Politecnico di Torino. Il workshop è stato pianificato per il secondo semestre dell'anno accademico, da marzo a giugno, con un appuntamento settimanale di 7 ore. La proposta del professore Mario Sassone comprendeva la collaborazione dello scrivente e dei colleghi del DAPe[3] arch. Tomas Ignacio Mendez Echenagucia, arch. Iasef MD Rian e arch. Shaghayegh Rajabzadeh. Data l'organizzazione dei workshop opzionali, era prevista l'iscrizione di un numero di studenti variabile dai 10 ai 40 al massimo. Il titolo del workshop, *Mor-*

[2]Le piattaforme in questione sono reperibili ai seguenti indirizzi: `http://areeweb.polito.it/didattica/spazioabitare/` e `http://areeweb.polito.it/didattica/cuscostudio/`.

[3]Il DAPe è la Scuola di Dottorato in Architettura e Progettazione edilizia attiva presso il DAD - Dipartimento di Architettura e Design del Politecnico di Torino.

fogenesi Computazionale, sottointendeva che l'argomento principale sarebbe stato lo studio particolareggiato di strumenti di calcolo digitale appositamente pensati per lo sviluppo e la definizione di forme nello spazio. L'aspetto computazionale, cioè derivato dall'uso di calcolatori per la risoluzione di problemi e calcoli normalmente inaccessibili per i tempi e le modalità di calcolo umane, viene quindi utilizzato per scopi di morfogenesi, ovvero di ricerca di forme. La morfogenesi computazionale è dunque: "un metodo di ricerca di forme tramite il supporto di algoritmi, lineari o genetici. Il software è così in grado di generare delle forme all'interno di certi limiti imposti, si prosegue poi alla verifica del loro comportamento (strutturale, acustico, etc.) e a eventuali ottimizzazioni e modifiche." (Pugnale 2007). Tali strumenti, nel caso dell'edizione 2013 del corso, avrebbero dovuti essere usati per la definizione di una struttura di muratura portante di forma libera, essendo questo uno dei temi di ricerca scientifica di alcuni dei promotori (il prof. Sassone e l'arch. Rajabzadeh in particolare) (Figura 6.1). Si è deciso che tale

Figura 6.1: La ricerca sulle volte in muratura di forma libera si inserisce in un più ampio contesto internazionale. Nell'immagine, i risultati del gruppo di ricerca presso l'ETH di Zurgio guidato dal professor Block (fonte: `http://block.arch.ethz.ch/brg/research/project/free-form-catalan-thin-tile-vault`).

prototipo avrebbe dovuto essere un piccolo padiglione di meditazione per il campus del Politecnico di Torino, al pari di altri padiglioni già esistenti in altri contesti simili[4]. Tale scelta è stata dettata in primo luogo dalla volontà

[4]Si fa qui riferimento alla Rothko Chapel a Houston ad opera di Philip Johnson, Howard Barnstone, Eugene Aubry and Mark Rothko; la MIT Chapel di Eero Saarinen a Cambridge; il Meditation Centre all'UNESCO di Tadao Ando a Parigi; il centro di me-

Figura 6.2: Uno dei riferimenti progettuali di BrickShell: il centro di meditazione e preghiera a Khartoum realizzatto da TAMassociati. La luce zenitale opportunamente filtrata è stato uno dei temi progettuali principali di BRickShell (fonte: `http://www.tamassociati.org/PAGES/PAD/PAD_PrayerPavilion.html#`).

di definire una funzione precisa ma non complessa (al fine di evitare di progettare un padiglione unicamente a scopo dimostrativo), e in secodno luogo per coinvolgere l'istituzione ospitante nel processo realizzativo, proponendo un manufatto che sarebbe potuto risultare di pubblica utilità. Si è deciso di chiamare il padiglione BrickShell, al fine di evidenziare le sua caratteristiche principali: la forma di un guscio, propria delle volte (Shell), e il fatto di essere realizzato in muratura (Brick).

Il workshop si è dato come obiettivo non solo quello di trasferire la conoscenza circa i più avanzati strumenti digitali a supporto della progettazione architettonica agli studenti coinvolti, ma anche di indagare, attraverso la costruzione di un prototipo, le possibilità di controllo del progetto in fase realizzativa consentite da tali strumenti.

---

ditazione presso il centro cardio-chirurgico Salm a Khartum in Sudan di TAMassociati (Figura 6.2); la cappella temporanea di St-Loup a Pompaples di Local Architecture.

6.1.1 Obiettivi, ipotesi iniziali e adozione dell'Open Source

All'inizio del corso gli studenti erano sedici.[5] Essendo il workshop pensato per le laureee magistrali, vi erano studenti provenienti da corsi differenti, con conoscenze e capacità diverse. Il workshop si era posto tre obiettivi ambiziosi: portare gli studenti a occuparsi di morfogenesi computazionale, progettare un edificio in volte di muratura di forma libera e, infine, svolgere un'esperienza di cantiere piuttosto intensa. Il tutto in circa quattro mesi, con un solo appuntamento settimanale. Considerando che gli studenti non avevano specifiche conoscenze sul tema e si doveva arrivare a realizzare un unico prodotto finale, si è pensato che sarebbe stato meglio instaurare un processo di tipo collaborativo in cui tutti gli studenti potevano partecipare cooperando, piuttosto che seguire un modello pedagogico di tipo concorrenziale, secondo il quale ogni studente (o gruppo di studenti) avrebbe dovuto sviluppare un progetto, e il progetto migliore sarebbe stato selezionato per essere poi realizzato. Infatti questo secondo tipo di approccio non sempre funziona, poiché nella successiva fase realizzativa, dopo la proclamazione di un vincitore, sono impegnati solo gli autori del progetto (a cui sta a cuore la corretta realizzazione) e non tutti gli altri partecipanti. Essendo la componente pratica uno degli obiettivi del corso, non sarebbe stato possibile limitarla a pochi partecipanti, ma era necessario trovare un modo per coinvolgere tutti gli studenti iscritti. Inoltre il carico di lavoro del workshop era piuttosto intenso e sarebbe risultato troppo oneroso in termini di tempo se ogni partecipante avesse dovuto lavorare in forma individuale. Si è dunque pensato che un sistema aperto, basato sul modello open source, potesse essere utile sia per quanto riguarda l'aspetto didattico (facendo riferimento alla learning organization cui si faceva cenno nel primo capitolo), sia dal punto di vista progettuale (in un sistema in cui i partecipanti hanno poco tempo e poca esperienza è meglio se gli forzi si uniscono invece che dividersi), sia dal punto di vista della costruzione (far affezionare tutta la comunità al progetto in modo che questo venga costruito collegialmente). BrickShell è quindi diventato un progetto di Architettura Open Source.

[5] Giulio Vianello, Emanuele Protti, Priscilla Thiesen Becsi, Gabriele Simonetta, Mariangela Rossino, Riccardo Pilleri, Silatchom Meguem Tadie, Santiago Jimenez, Simone Iennarella, Andrea Galli, Joline Folle, Fulvio Brunetti, Wilian Destefani, Tito Ceci De Sena, Carlos Henrique De Assis, Giovanni Bouvet.

6.1.2 Tema e campo di applicazione

Il tema di questa particolare esperienza è quello delle volte di forma libera in muratura. Sul tema delle volte è possibile trovare edifici di vario tipo e con funzioni diversi ma è stato scelto di realizzare un edificio di uso e utilità pubblica all'interno del Politecnico di Torino. Il campo di applicazione principale è la didattica, un campo di applicazione che non era stato ancora esplorato dai casi studio presi in esame, i quali, come abbiamo visto, si sono concentrati su altri obiettivi. Tuttavia con essi viene condiviso l'aspetto informale dell'iniziativa, poichè la natura sperimentale del workshop ha fatto in modo che tutta l'esperienza non potesse essere inquadrata nella normale attività edilizia ma avesse uno status particolare e non normato, sperimentale per l'appunto.

6.1.3 Definizione della sorgente

> *"Open-endedness, variability, interactivity, and partecipation are the technological quintessence of the digital age. They are here to stay And soon designers will have to choose. They may design objects, and then be digital interactors. Or they may design objectiles, and then be digital authors. The latter choice is more ardous by far, but its reward are greater."*[6]
>
> (Carpo 2011, p. 126)

Il workshop Morfogenesi Computazionale si è basato sulla definizione di algoritmi. Questi possono essere descritti, secondo quanto affermato da David Knuth, uno dei padri dell'informatica, come "una sequenza ben definita di regole che ci dicono come produrre o calcolare un'informazione, in base a un insieme di dati precedentemente assegnati, in un numero finito di passi". Per la compilazione di tali algoritmi viene utilizzato solitamente un linguaggio di programmazione e la sintassi che il linguaggio scelto prevede. Per poter scrivere un algortimo non per forza è necessario conoscere un linguaggio di programmazione e la sua sintassi, poichè alcuni programmi (e alcuni di questi sono programmi cad 2d/3d) hanno implementato degli strumenti di programmazione visuale. Ciò significa che invece che scrivere ogni singola funzione di un algoritmo attraverso un linguaggio di programmazione, è possibile ottenere lo stesso risultato tramite la manipolazione grafica degli elementi e non

[6] "L'apertura infinita, la variazione, l'interattività e la partecipazione sono la quintessenza tecnologica dell'età digitale. Non sono un fenomeno passeggero. E presto i progettisti dovranno scegliere. Potranno disegnare oggetti, e in seguito interagirvi attraverso il digitale. O protranno disegnare degli objectiles, e diventare degli autori digitali. La seconda scelta è più difficile, ma la ricompensa è maggiore." [traduzione italiana a cura dell'autore].

tramite sintassi scritta. Uno strumento di programmazione visuale consente di programmare con espressioni visuali, essendo basato sull'idea 'boxes and arrows' [7], dove le 'boxes' (scatole contenti oggetti, parametri o funzioni) sono collegate logicamente tra di loro attraverso le 'arrows' (letteralmente frecce, ovvero i collegamenti logici tra le parti).

La sorgente, essendo il workshop focalizzato sulla morfogenesi computazionale, è costituita da un algoritmo (o un insieme di più algoritmi). Ciò significa che invece di avere un disegno di una forma o di un oggetto, o un modello tridimensionale, si hanno delle istruzioni e funzioni che portano alla definizione di quella forma. L'insieme di queste istruzioni dà luogo a un modello parametrico, un modello la cui forma può essere variata agendo su opportuni parametri (parametri dimensionali, ambientali, strutturali, normativi o altro). Il modello parametrico diventa dunque la sorgente del progetto BrickShell.

La natura di questa sorgente presenta due modalità di interazione piuttosto diverse. La prima modalità, pensata per utenti esperti, è rivolta alla definizione della sorgente. Attraverso l'uso del software appropriato è possibile definire la sorgente e, di conseguenza, la forma di BrickShell, agendo direttamente sul modello parametrico, scegliendo cioè quali parametri includere, quali funzioni utilizzare e quali solo le relazioni logiche che intercorrono tra di loro. Questo lavoro assomiglia molto al lavoro che viene fatto normalmente sul software, dove le varie funzionalità del software vengono via via implementate, migliorate, semplificate, aggiunte o tolte a seconda delle necessità.

La seconda modalità di interazione avviene quando si agisce sulla variazione dei parametri che definiscono alcune caratteristiche principali del modello. Agendo sui parametri, la sequenza logica dell'algoritmo non viene cambiata ma si modifica il risultato finale. Si può, ad esempio, variare la scala di un oggetto, variarne l'altezza e la larghezza, senza però riscrivere le regole che portano alla definizione di quell'oggetto, ma agendo unicamente sulla variazione dei valori dei parametri. Le due modalità sono ugualmente importanti poiché entrambe permettono di adeguare l'oggetto finale alle proprie necessità, seppur necessitando di operazioni e conoscenze di base differenti.

La sorgente di BrickShell non comprende solo la sua definizione formale, ma anche molte delle informazioni relative alla sua costruzione. Trattandosi di un edificio di volte in muratura, la sorgente include anche la definizione delle centine per la sua costruzione e, a livelli più avanzati, potrebbe anche comprendere tutte le informazioni relative alla posizione e alla forma di ogni singolo mattone, qualora il singolo mattone dovesse essere oggetto di aggiu-

[7] http://it.wikipedia.org/wiki/Linguaggio_di_programmazione_visuale

staggio (trattandosi di superficie di forma libera è lecito aspettarsi che un pattern bidimensionale costante nelle dimensioni come quello della muratura non possa essere applicato uniformemente sulla superficie ma necessiti di qualche modifica puntuale). Da tali considerazioni si evince che la sorgente sarà oggetto di continuo sviluppo e implementazione.

6.1.4 Definizione della comunità

> *"Se pensate che nel vostro gruppo di lavoro ci sia qualcuno che ha un'idea migliore della vostra, perché non accoglierla e svilupparla? Il risultato finale sarà diverso, e il confronto prezioso."*
>
> (Potter 2002, p. 172)

> *"When you discuss your own work you have to ask yourself what you acquired from whom. Because everything you find comes from somewhere."*[8]
>
> (Hertzberger 1991, p. 5)

L'obiettivo principale del workshop è naturalmente un obiettivo didattico. Essendo un'esperienza didattica esiste una comunità predefinita formata da diversi utenti, i quali si possono suddividere in studenti (utenti sviluppatori) e docenti/tutor (utenti promotori). Un gruppo limitato di studenti è quindi la comunità di base. Questa limitazione può avere degli effetti negativi: la natura aperta dell'iniziativa non toglie la possibilità che altri utenti si aggiungano alla comunità, ma il tipo di esperienza proposto è rivolto ad un gruppo molto specifico di persone. Gli utenti possibili sono infatti degli architetti o degli studenti di architettura, e, se studenti, tendenzialmente delle lauree magistrali vista la complessità dei temi svolti, già competenti per quanto riguarda la morfogenesi computazionale e, dato l'oggetto dell'iniziativa, tendenzialmente affiliati al Politecnico di Torino. L'aspetto positivo di questa limitazione è che una comunità limitata di utenti è facilmente monitorabile e le dinamiche che intercorrono tra i vari partecipanti possono essere registrate e analizzate agevolmente. Questo secondo aspetto non è da sottovalutare poiché BrickShell non è solo un workshop didattico, ma anche un esperimento scientifico sui seguenti aspetti: l'Architettura Open Source, la tecnica di costruzione di volte in muratura di forma libera, e infine, l'utilizzo di avanzati strumenti di morfogenesi computazionale. La replicabilità è quindi alla base di questa esperienza, la quale vuole essere un modello per un possibile utilizzo diffuso in futuro.

[8] "Quando analizzate un vostro lavoro dovete domandarvi che cosa avete acquisito e da chi. Poiché tutto ciò che sviluppate arriva da qualche altra parte." [traduzione italiana a cura dell'autore].

6.1.5 Definizione della piattaforma

> *"With increasing globalization and specialization in the design and building industry, collaboration between partners in remote locations becomes crucial. Ideally, all of them could work on a building design at any place, simultaneously together (synchronously) or separately (asynchronously), while the latest state of the design would always be available in a shared database to all team members. They could collaborate on a shared object and no information would thus be lost in transfer of project data. But to be successful, this emerging type of collaboration often requires new design and communication methods"*[9]
>
> (Schmitt, Hirschberg, Kurmann, Kolarevic, Johnson & Donath 1999b)

Per quanto riguarda la piattaforma ci si è dovuti confrontare con alcune limitazioni che si sono presentate all'inizio dell'esperienza. La prima era dovuta alle risorse scarse di cui si disponeva per definire una piattaforma ad hoc che potesse essere usata in modo proficuo dalla comunità. Una volta deciso di adottare l'Open Source come metodo di sviluppo, non si disponeva però del tempo necessario per poter progettare e realizzare un sito web che potesse ospitare un lavoro di tipo collaborativo senza che questa diventasse anch'essa una operazione di tipo sperimentale. Di conseguenza non era possibile avere una piattaforma stabile e di semplice utilizzo, con la possibile aggravante di caricare gli utenti di una frustrazione non necessaria (dovuta al funzionamento non fluido della piattaforma), rischiando così il fallimento dell'iniziativa. Infatti in altre esperienze didattiche che hanno fatto uso di piattaforme di collaborazione disegnate ad hoc (gli atelier *Lo spazio dell'abitare* tenuti dal professori P.A. Croset ed E. Piccoli presso il Politecnico di Torino a partire dal 2011) ci si era scontrati con una certa difficoltà iniziale di utilizzo da parte di utenti non esperti, difficoltà che era stata superata col tempo ma in condizioni di utilizzo relativamente rilassate e in cui la piattaforma non era lo strumento principale di lavoro ma uno strumento di accompagnamento.

Oltre alla mancanza di risorse per la sperimentazione, era da tenere in considerazione la poca familiarità degli utenti con gli strumenti proposti, fa-

[9] "A seguito della crescente globalizzazione e specializzazione nel campo della progettazione e costruzione degli edifici, la collaborazione tra colleghi in luoghi anche distanti tra loro diventa cruciale. Idealmente, tutti loro potrebbero lavorare sullo stesso progetto pur stando in posti differenti, simultaneamente o in momenti diversi, in modo che l'ultima versione del progetto sia sempre disponibile in un database condiviso a tutti i membri del gruppo di progettazione. Possono collaborare su di un progetto condiviso senza che alcuna informazione venga persa durante i trasferimenti di dati. Ma per fare ciò con successo, questo nuovo metodo di collaborazione richiede nuovi metodi di progettazione e di comunicazione." [traduzione italiana cura dell'autore].

miliarità che doveva invece essere coltivata con un certo sforzo nei confronti dei software di progettazione parametrica. La definizione di una piattaforma disegnata appositamente per BrickShell non è stata dunque una priorità, ma ciò non ha impedito che si potessero comunque mettere a punto altri strumenti altrettanto validi per ottenere una piattaforma funzionante e performante.

Si è quindi scelto di suddividere la piattaforma in diversi servizi e, per ogni servizio, di utilizzare uno strumento già esistente modificandolo secondo le proprie necessità, tentando di utilizzare strumenti che fossero di grande diffusione di modo da superare anche l'iniziale diffidenza degli utenti più recalcitranti.

Per il repository ufficiale e il sito web principale si è deciso di appoggiarsi a una piattaforma esistente di cui in precedenza si è parlato lungamente: l'Open Architecture Network. Come già segnalato, questa piattaforma risulta essere disponibile a ospitare progetti aperti di vario genere, che possono trovare un luogo dove avviare la propria attività senza particolari risorse iniziali. La pagina di BrickShell sull'Open Architecture Network diventa dunque il sito web principale e il repository dell'iniziativa[10]. La pagina è visitabile da parte di chiunque ma può essere modificata solo dagli utenti registrati e il materiale è liberamente scaricabile da chiunque. Inoltre OAN permette di utilizzare licenze Creative Commons per tutto ciò che viene condiviso.

Per quanto riguarda invece la comunicazione e il confronto tra gli utenti è stato scelto di utilizzare un network sociale di utilizzo diffuso di modo che non ci fossero problemi o difficoltà di utilizzo, come Facebook [11], nella convinzione che la diffusione presso gli studenti dell'ateneo (e non solo) potesse favorire l'accesso da parte di nuovi utenti. Facebook dà la possibilità a chiunque di realizzare una pagina dedicata a un singolo argomento, e di fare in modo che sia possibile condividere vari materiali in maniera piuttosto facile e diretta. Un ulteriore strumento utilizzato è stato Dropbox[12], uno strumento personale di archiviazione e condivisione di dati. Il software in questione, installato sul computer di ciascun utente, permette a chi lo utilizza di poter conservare dei dati sui server messi appositamente a disposizione e di potervi accedere da diversi computer o, addirittura, di poter condividere con altri utilizzatori intere cartelle di file. Dà inoltre la possibilità di ottenere un link per ciascun file caricato, link che può essere facilmente condiviso e che risulta essere sempre attivo. Ciò permette agli utenti di poter condividere dei file di lavoro attraverso la piattaforma Facebook in totale sicurezza e relativa

[10] `http://openarchitecturenetwork.org/projects/BrickShell`
[11] `https://it-it.facebook.com/pages/BrickShell/155528374601155`
[12] `https://www.dropbox.com/`

facilità unicamente postando il link del file interessato. In ultimo è stato utilizzato anche Google Drive [13], un servizio gratuito messo a disposizione da Google che permette a più utenti di lavorare su documenti condivisi come ad esempio file di testo o tabelle di calcolo.

Inoltre vi sono gli strumenti che normalmente vengono utilizzati per operare direttamente sulla sorgente. Questi sono principalmente due: Rhinoceros[14] e Grasshopper[15]. Il primo è un software di modellazione tridimensionale tramite NURBS, ovvero Non Uniform Rational B-Spline, un metodo algoritmico per la costruzione di curve e superifici free-form, molto utile se non indispensabile in questo caso. Il secondo è un plugin del primo, ovvero un software che si attiva all'interno del più complesso Rhinoceros, e che offre un sistema di programmazione visuale. Permette cioè di utilizzare le funzioni derivanti dai comandi di Rhinoceros come oggetti visuali, e di collegarli logicamente a parametri e oggetti disegnati. Se il primo può essere scaricato in versione di prova, il secondo può essere scaricato liberamente. La versione di prova di Rhinoceros permette unicamente 25 salvataggi, dopodiché continua a funzionare ma non permette di salvare. Dal momento che però i file vengono creati unicamente con Grasshopper, la limitazione di Rhinoceros non costituisce un problema.

6.1.6 Svolgimento

Preparazione

> *"The preparation of a collaborative project naturally must be a collaborative effort in itself."*[16]
>
> (Schmitt, Hirschberg, Kurmann, Kolarevic, Johnson & Donath 1999a)

Il primo mese del workshop è stato dedicato a diverse attività propedeutiche. La maggior parte degli sforzi dei partecipanti si sono concentrati sull'apprendimento dei metodi di progettazione parametrica. Parallelamente a questo tipo di approfondimento vi è stato lo studio delle tecniche di costruzione delle volte e della scienza delle costruzioni. Oltre a queste componenti c'è stata anche l'introduzione delle varie componenti della piattaforma, nella convinzione che esattamente come si può imparare a realizzare e gestire un

[13] `http://www.google.com/intl/it/drive/about.html`

[14] `http://www.rhino3d.com/`

[15] `http://www.grasshopper3d.com/`

[16] "La pianificazione di un progetto collaborativo deve naturalmente essere anch'essa una operazione collaborativa." [traduzione italiana cura dell'autore].

modello parametrico si possa imparare a collaborare efficacemente con altre persone.

Sono state messe a punto alcune piccole esercitazioni che potessero permettere agli utenti di utilizzare la piattaforma e di collaborare come in una comunità avviata. Questi esercizi sono stati pianificati seguendo il modello di quanto era stato fatto già nel 24 Hour Design Cycle (Schmitt et al. 1999a) dai professori Schmitt e Hirschberg dell'ETH di Zurigo. Le esperienze svolte presso l'ateneo svizzero prevedevano infatti risoluzioni cooperative a problematiche progettuali attraverso un uso collaborativo della rete (siamo nel 1999, ancora lontani dal web 2.0). Il lavoro del 24 Hour Design Cycle si basava sul modello della staffetta: prevedeva il coinvolgimento di diverse scuole di architettura su scala mondiale (Zurigo, Shanghai e Seattle) e l'obiettivo era quello di far lavorare gruppi di studenti di diversa provenienza come in una staffetta a tempo, con turni equivalenti e con un scambio continuo del progetto, il quale procedeva nella sua definizione insieme al lavoro svolto dai vari gruppi (Figura 6.3). Questo modello di collaborazione, legato alla successio-

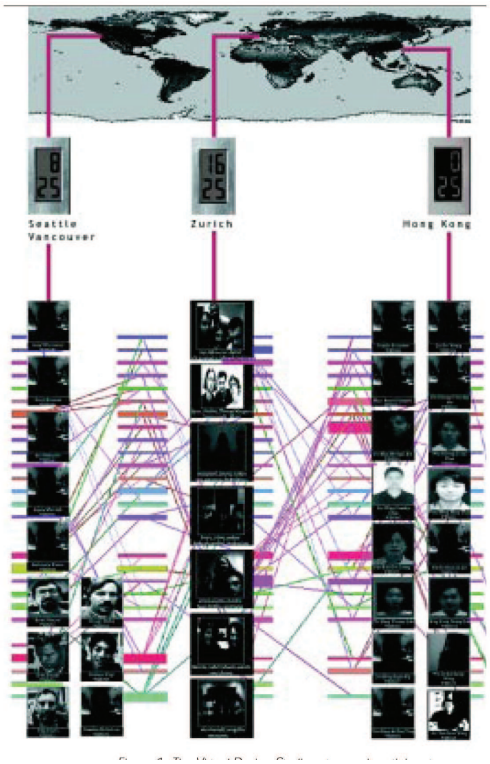

Figura 6.3: L'esperienza del 24 hour Design Cycle viene rappresentata in questo schema: i gruppi d iprogettazione si sono scambiati i file di lavoro durante 24 ore tra le tre sedi coinvolte. L'immagine è tratta da Schmitt et al. (1999b).

ne temporale e logica dei turni di lavoro, pemette un avanzamento costante e un riconoscimento necessario del lavoro del proprio predecessore. Risulta molto efficace per esperienze che devono essere svolte in tempi ristretti (ad esempio OpenSimSIm con l'esperienza OpenJapan ha adottato lo stesso sistema) e costringe ogni utente a confrontarsi con il lavoro e con le posizioni dei propri colleghi.

In questo senso, la prima esercitazione ha previsto la compilazione di un racconto collaborativo. Dato l'inizio e la fine del racconto e un elenco di turni di scrittura, ogni utente è stato chiamato, seguendo l'ordine prestabilito, a scrivere una parte intermedia del racconto seguendo ciò che era stato scritto precedentemente dal proprio compagno. L'ultimo utente si sarebbe dovuto poi collegare alla fine della narrazione già presente nel documento. Effettuata lungo la prima settimana di lavoro, questa esercitazione ha previsto l'uso di Google Drive per la scrittura condivisa e di Facebook per la condivisone di quanto scritto. L'esperimento ha avuto successo: tutti gli utenti coinvolti hanno partecipato e ognuno ha contribuito a sviluppare una parte del racconto.

Il secondo esercizio prevedeva lo sviluppo di un semplice algoritmo attraverso Grasshopper. Organizzato nella stessa maniera dell'esercitazione precedente, ogni utente ha utilizzato Rhinocerso e Grasshopper e condiviso il lavoro con gli altri utenti via Facebook utilizzando Dropbox. In questa esercitazione gli utenti dovevano contribuire a comporre una definizione di algoritmo attraverso l'uso di Grasshopper e per fare ciò, ogni utente poteva aggiungere due elementi, sostituire un elemento o eliminare due elementi già presenti nel file. Dopo queste operazioni il file doveva essere passato all'utente successivo, il quale poteva a sua volta effettuare le operazioni che aveva a disposizione e così via. Si è partiti da un file molto semplice con qualche parametro per arrivare a una definizione più complessa e a un prodotto finale pressoché completo. Anche questo esercizio ha avuto successo, nel senso che l'algoritmo è stato sviluppato, l'elenco è stato seguito e qualche utente ha continuato a sviluppare l'algoritmo anche dopo la fine dell'esercitazioneo.

Il terzo esercizio si è focalizzato sull'utilizzo di Grasshopper: data una definizione di algoritmo piuttosto complicata, ogni utente era chiamato a effettuare operazioni di semplificazioni. Per fare ciò si sono seguiti gli stessi principi che governavano l'esercitazione precedente e anche gli stessi strumenti.

Sviluppo collettivo della sorgente Terminata la fase di preparazione gli utenti erano ormai in grado di gestire un modello parametrico, di utilizzare la piattaforma messa a loro disposizione, e si erano abituati a lavorare in modo

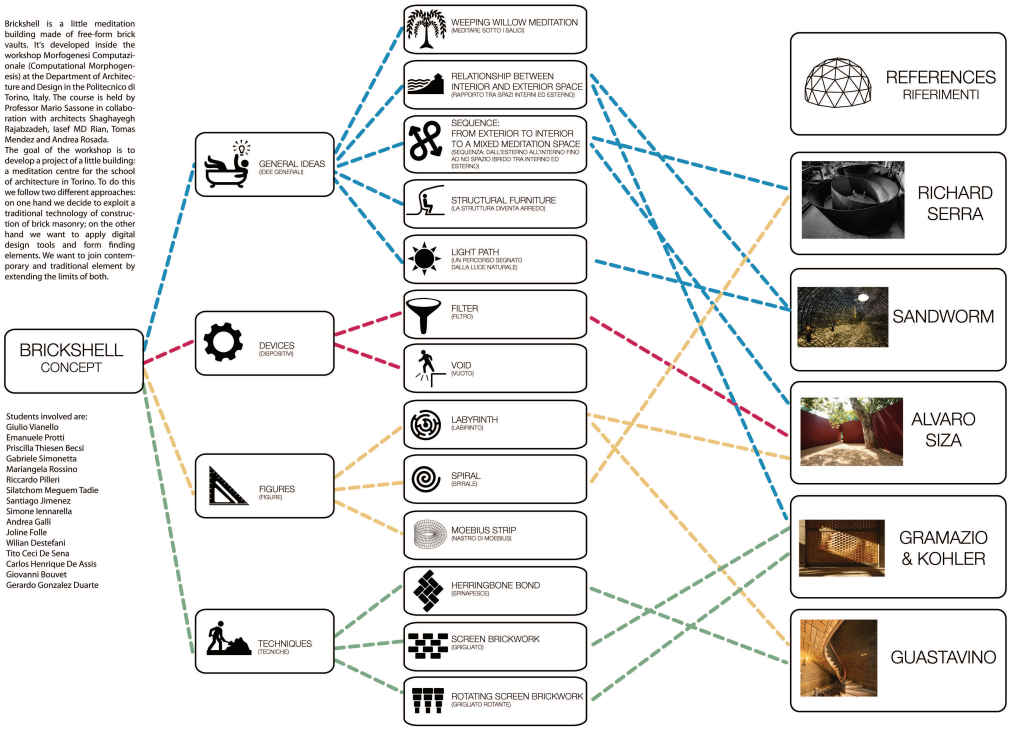

Figura 6.4: La versione 0.00 di BrickShell: il diagramma.

collaborativo. Si è quindi iniziato il lavoro sulla sorgente. Tale lavoro, svolto in maniera collaborativa, ha portato alla definizione della prima effettiva sorgente progettuale da cui è stato sviluppato il modello parametrico. Come si è detto in precedenza l'oggetto progettuale è stato un padiglione di meditazione per il campus del Politecnico di Torino. Sulla scorta di esempi già citati in precedenza, la comunità di BrickShell è stata chiamata a progettare e realizzare un centro di meditazione per l'Ateneo torinese. In un periodo di circa due settimane si è quindi ragionato, e attraverso la piattaforma online e attraverso gli incontri settimanali sui caratteri che questa nuova architettura avrebbe dovuto includere, sia partendo da riferimenti di tipo architettonici, che da riferimenti diversi di tipo artistico, costruttivo e naturale. Il risultato di questa prima fase di lavoro è stato sintetizzato in un diagramma contenente tutti gli elementi condivisi a cui la comunità avrebbe potuto attingere per il suo lavoro futuro (Figura 6.4).

Il diagramma presenta quattro principali ambiti di indagine: idee generali, dispositivi spaziali, figure geometriche e tecniche costruttive. A queste quattro componenti vengono collegati alcuni dei riferimenti principali presi in esame. Le idee generali comprendono tutti quei caratteri propri di un centro di meditazione che la comunità ha deciso di implementare: si tratta

perlopiù di sensazioni che il risultato finale dovrebbe suscitare nel visitatore o di modalità di fruizione e di funzionamento dell'edificio. All'interno dei dispositivi spaziali sono comprese alcune soluzioni che, se utilizzate, garantirebbero alcuni degli effetti auspicati dalle idde generali. Nel caso delle figure si tratta di configurazioni geometriche esistenti che possono essere prese da esempio. Nella voce 'tecniche costruttive' sono annoverate alcune delle possibili disposizioni dei mattoni che possono essere utilizzate efficacemente per realizzare la volta in forma libera di muratura. Il diagramma è diventato la prima sorgente progettuale da cui la comunità è partita per costruire il modello parametrico.

Se normalmente la sorgente viene definita a priori dai promotori dell'iniziativa, in BrickShell si è preferito procedere alla definizione di una sorgente condivisa, considerando che l'utilità pubblica e la particolare funzione dell'oggetto da costruire necessitassero di un processo di definizione collaborativo. L'unico elemento definito a priori dai promotori è stata la componente costruttiva, ovvero il fatto che l'edificio finale dovesse necessariamente essere realizzato in volte di forma libera in muratura.

Metodo di sviluppo collaborativo Il metodo 'a staffetta' di Schmitt e Hirschberg, che è si è dimostrato molto efficace in una fase iniziale, non è stato più adottato in seguito, cioè nel momento in cui si è iniziato a sviluppare la sorgente in modo aperto.Tale scelta è motivata dalle seguenti considerazioni. La prima è che la definizione di turni non permette il facile inserimento di nuovi utenti. In un sistema completamente aperto un utente può prendere liberamente la sorgente, apportare le sue modifiche secondo il suo interesse e la sue necessità, presentare i risultati alla comunità e di lì in poi si attua quel fenomeno organizzativo basato sulla reputazione di cui si è parlato abbondantemente in precedenza. Invece in un sistema troppo rigido di turni ciò non potrebbe avvenire, sia perché il giudizio sull'operato del nuovo utente sarà emesso unicamente dal suo successore nella lista predefinita, sia perché non vi è un sistema per definire quale utente può essere inserito nella lista e quale no. Quindi un sistema libero sembra più efficace, proprio perché che un sistema di turni limita molto l'attività della comunità. Inoltre se è vero che un un sistema a turni può funzionare bene in comunità molto ridotte e già predefinite (come può essere la comunità di un gruppo di studenti di uno specifico corso), va aggiunto però che il fattore temporale, che stabilisce l'efficacia di un sistema di turni in un tempo ben definito e comunque ridotto, limiterebbe il workshop a qualche giorno. In un sistema più flessibile, che si protrae magari per qualche mese o anno, non è detto che il sistema a turni funzioni altrettanto bene. Bisogna infatti considerare anche altri elementi:

l'Open Source per come lo conosciamo è un sistema di lavoro volontario, non remunerato in forma economica diretta (attraverso una transazione monetaria) ma attraverso altri canali, come ad esempio l'esperienza acquisita, la risoluzione a basso costo di una istanza personale, il semplice hobby creativo. Come tale difficilmente può essere inquadrato in un sistema di lavoro organizzato per turni obbligatori. Come affermava Stallman ("free as in freedom"), la libertà sta alla base del sistema, il che permette a ogni utente di intervenire come e quando gli è più congeniale, favorendo un utilizzo più efficace della 'risorsa' utente secondo i tempi e i modi definiti da ogni singolo partecipante. Questo sistema, che a prima vista può sembrare confusionale, ha in realtà dimostrato la sua efficacia in quasi tutte le iniziative legate all'Open Source. È vero che l'assenza di turni può generare due o più versioni 'parallele' contrastanti, poiché è possibile che due utenti lavorino contemporaneamente sulla sorgente ma sviluppino soluzioni diverse. Tuttavia ciò non risulta essere un problema dal momento che in maniera autonoma una delle versioni, probabilmente la più valida o quella che la comunità ritiene essere la più valida, verrà ulteriormente sviluppata mentre le altre verranno abbandonate.

Si è deciso dunque di lasciare gli utenti liberi di gestire lo sviluppo della sorgente in maniera assolutamente autonoma, adottando come unico sistema di organizzazione quello del versioning.

Versioning Il versioning è il metodo attraverso il quale le varie versioni progressive di un software vengono nominate. Solitamente si tratta di un sistema di numerazione che permette di identificare alcune 'major release' (versioni complete del software che comprendono una serie di funzionalità) e le relative 'minor release' (versioni successive a una major release il cui scopo è quello di affinare il funzionamento della stessa e risolverne i malfunzionamenti). Il versioning viene realizzato attraverso una numerazione progressiva, a partire dallo zero. Ogni versione è indicata attraverso l'utilizzo di due numeri separati da un punto. Il primo numero indica la major release, il secondo la minor release. La prima versione si chiamerà quindi versione 0.0, le sue minor release 0.1, 0.2, 0.3 e così via, fino ad arrivare alla seconda versione, la 1.0. Al versioning è dovuta anche la denominazione web 2.0, ovvero una versione del web più avanzata rispetto quella precedente.

Lo stesso sistema è stato usato anche per la comunità di BrickShell. A una major release principale (la cui forma e i cui contenuti sono stati definiti negli incontri settimanali in maniera collegiale) sono corrisposte diverse minor releases durante il corso della settimane. Tali minor releases sono state prodotte da ciascun utente secondo le sue particolari capacità. La numerazione ha aiutato il resto della comunità a orientarsi all'interno del materiale

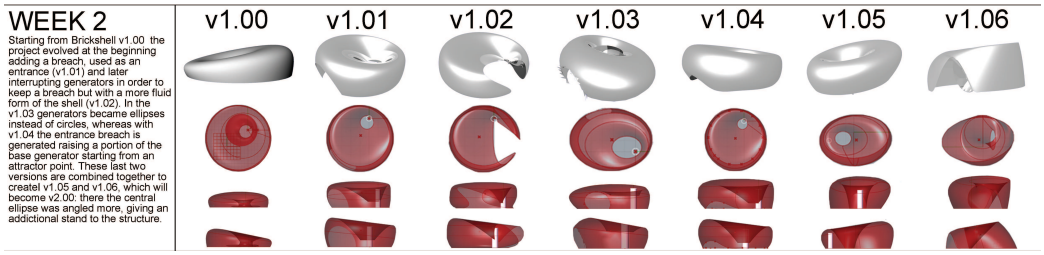

Figura 6.5: L'evoluzione della versione 1.00.

prodotto. Ciascuna major release è stata resa disponibile sulla piattaforma OAN ogni settimana, per un totale di 8 settimane. Ciò ha permesso di stabilire uno stato dell'arte settimanale su cui è stato possibile lavorare in seguito. Durante la settimana ogni utente ha lavorato liberamente e portato avanti il lavoro in piena autonomia; non appena raggiunto qualche progresso significativo si è condiviso con il resto della comunità l'esito del lavoro. La comunità poteva osservare il lavoro svolto e se un altro utente valutava che la sorgente si stava sviluppando in maniera positiva, poteva continuare il lavoro iniziato in precedenza. In caso di più versioni 'parallele' è stata la versione reputata migliore ad essere automaticamente sviluppata, mentre le altre sono rimaste in uno stato di 'work in progress' e abbandonate nel momento in cui una nuova versione ufficiale è stata rilasciata.

Data la natura della sorgente è anche possibile che le modifiche apportate non stravolgano la forma finale prodotta dallo script, ma che siano modifiche migliorative sullo script stesso per aumentare l'efficacia, la velocità e l'accessibilità, ovvero facilitarne l'uso da parte di altri. Lo script di Grasshopper tende infatti a essere territorio di sperimentazione continua e non sempre le soluzioni trovate, seppur riuscendo a raggiungere l'obiettivo preposto, sono le migliori dal punto di vista della programmazione. Si tratta cioè di script ridondanti e complessi che possono essere in seguito semplificati. Questo lavoro di semplificazione può essere oggetto di una nuova versione da parte di un utente, analogamente a quanto avviene nello sviluppo del software, dove ad ogni major release corrispondono svariate minor releases il cui scopo è quello di riparare i bachi esistenti.

Analisi dello sviluppo Lo sviluppo della sorgente è avvenuto seguendo rilasci settimanali e lavorando sulla sorgente il resto del tempo. Dopo ciascun rilascio, sono stati identificati gli obiettivi del rilascio successivo di modo che in capo a un paio di mesi si potesse ottenre un prodotto finito pronto per essere costruito.

Le versioni dalla 1.0 alla 4.0 hanno visto la genesi della forma e la costruzione di un modello parametrico che fosse in grado di definire efficacemente la forma desiderata (Figura 6.5). Questa fase comprende contributi di diverso tipo: da un lato un lavoro sulle varie parti e funzioni del modello, dall'altro un lavoro sui parametri e sulla loro variazione. Attraverso la versione 5.0 e 6.0 è stata definita la forma finale di BrickShell con un'attenzione alle problematiche costruttive: in queste versioni sono state introdotte le centine per la costruzione e gli strumenti di controllo strutturale. La versione 7.0 è il risultato del lavoro sulla componente strutturale e sul controllo della curvatura in ogni punto della superficie. In particolare il controllo della curvatura (e di conseguenza l'analisi strutturale) è stato condotto a modello ultimato, agendo unicamente sui parametri. Attraverso la variazione dei parametri si è infatti riusciti a ottenere valori puntuali di curvatura accettabili su tutta la superficie. La versione 8.0 ha visto ultimato il lavoro sulle centine rendendo disponibili i profili di taglio per la costruzione delle stesse (Figura 6.6). Nel

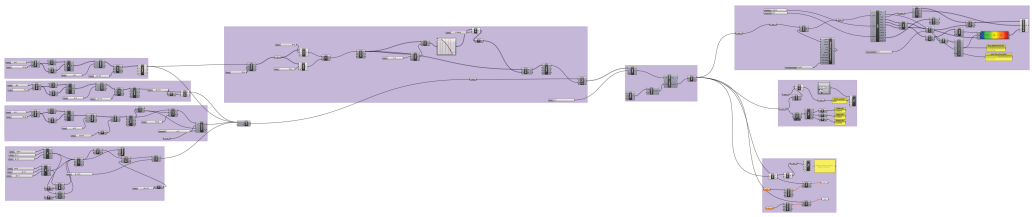

Figura 6.6: La versione 8.00 della sorgente di BrickShell: l'algoritmo realizzato con Grasshopper. A partire da questa versione sono state realizzate le centine in cartone.

complesso sono state rilasciate 8 versioni (più la versione 0.00, il diagramma) del modello parametrico su un totale di 8 settimane. La comunità ha partecipato attivamente a tutte le versioni. In totale sono state sviluppate circa quaranta versioni intermedie distribuite tra le 8 settimane. Solo nelle prime settimane si è verificata la presenza di versioni 'parallele', ma nonostante questo il processo di sviluppo non si è fermato: la comunità ha continuato scegliendo di sviluppare sempre una delle proposte piuttosto che altre. La maggior parte delle versioni intermedie sono state sviluppate nelle prime settimane poiché la mole di lavoro era decisamente più alta. Nelle settimane successive il lavoro sui parametri e sulle centine ha permesso di avanzare con meno versioni intermedie. Su un gruppo totale di 16 persone hanno lavorato attivamente sulla sorgente (proponendo cioè versioni intermedie) circa 10 utenti coinvolti, mentre il resto del gruppo ha sviluppato tutti i materiali corredati, principalmente materiali esplicativi e grafici.

Dal bit all'atomo Per quanto riguarda la costruzione, ovvero il passaggio da informazione digitale a materia fisica, si è proceduto all'organizzazione di un cantiere di tipo sostanzialmente tradizionale. Trattandosi di volte di forma libera, cioè di geometrie difficilmente definibili attraverso strumenti tradizionali, si è dovuto procedere alla costruzione di adeguate centine in grado di fungere da punti di appoggio e guida per la corretta posa dei mattoni. Attraverso la costruzione delle centine si è riusciti a trasferire la complessa informazione digitale riguardante la forma dell'edificio in supporto provvisionale per la costruzione, evitando di compiere errori di posa e riuscendo a rimanere all'interno delle tolleranze imposte dalla natura stessa dell'edificio. L'intera centina è stata pensata come un calco a scala reale della forma desiderata, realizzata attraverso l'utilizzo di fogli di cartone ondulato rigido. Nel modello digitale l'intero volume coperto dall'edificio è stato sezionato da serie di piani verticali nelle due direzioni principali: la distanza tra i piani è stata fissata a 20 cm. Ogni piano di sezione è stato poi sagomato secondo l'andamento della volta sezionata: l'incastro dei vari piani sagomati tra di loro ricostruisce la sagoma dell'intero edificio. Per facilitarne la costruzione, le sagome sono state realizzate in cartone ondulato e l'intera centina è stata suddivisa in 20 moduli di base quadrata che, accostati gli uni agli altri, formano la sagoma completa. La distanza tra le sagome non è stata casuale: bisogna considerare infatti che, essendo il mattone lungo 30 cm e venendo posato non per corsi orizzontali al terreno ma a spina di pesce, due piani distanti tra loro 20 cm offrono al mattone uno o due punti di appoggio, facilitando la posa in opera dello stesso (Figura 6.7). Per disegnare sui cartoni le sagome corrette di ciascun modulo si è inizialmente pensato di procedere utilizzando una taglierina a controllo numerico. Il modello parametrico forniva le sagome corrette di ciascun modulo, si trattava dunque di trasferire l'informazione dal computer alla macchina e si sarebbero ottenute tutte le sagome pronte per l'assemblaggio. Tuttavia data la mancanza di risorse economiche (il macchinario non era presente presso l'istituzione ospitante e ci si sarebbe dovuti rivolgere a un laboratorio esterno) si è deciso di procedere con gli strumenti disponibili. Presso il laboratorio LATEC del Politecnico di Torino, diretto dall'arch. Angela Lacirignola, è stato installato un proiettore attraverso cui sono state proiettate tutte le sagome. L'immagine proiettata, convenientemente collimata e scalata (l'utilizzo di una griglia di riferimento disegnata sul piano su cui veniva proiettata l'immagine della sagoma, e di una griglia di riferimento propria del file proiettato, e la conseguente operazione di collimazione tra le due griglie hanno permesso di ottenere immagini proiettate a scala reale) è stata riportata manualmente sui cartoni ondulati e poi ritagliata (Figura 6.8). Ogni pezzo è stato etichettato e si è poi proceduto con l'assemblaggio dei moduli. La muratura è stata realizzata da tutti

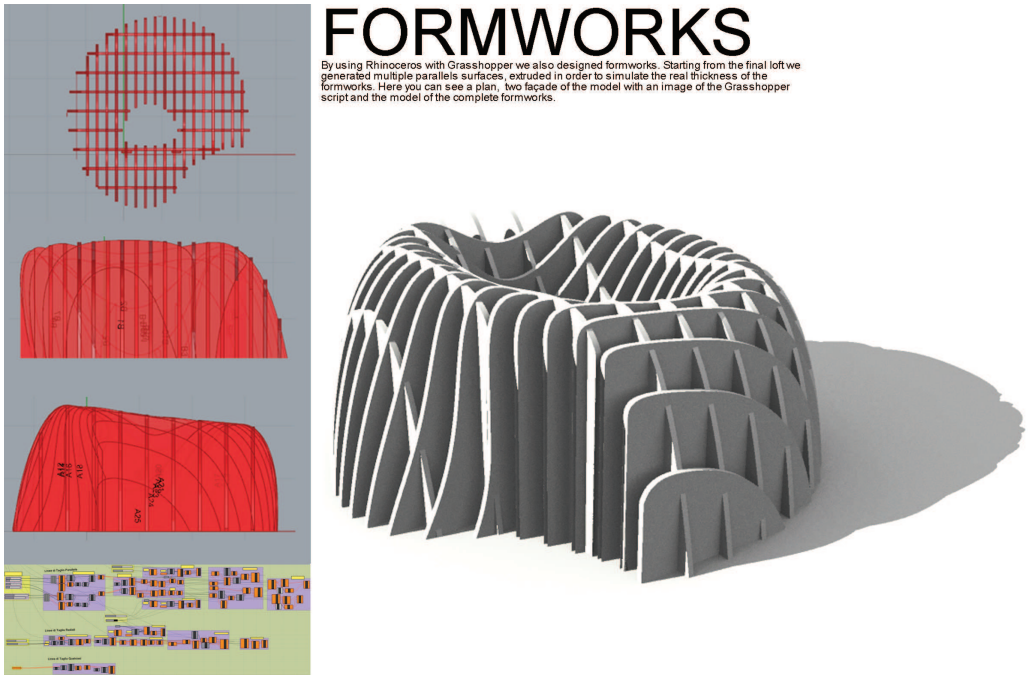

Figura 6.7: Uno schema esplicativo del metodo di definizione delle centine.

i paretecipanti nelle settimane successive la realizzazione delle centine. La costruzione è stato un momento in cui è stato possibile ampliare la comunità iniziale poichè l'attività di tipo ludico/costruttivo ha favorito la partecipazione di altri utenti che inizialmente non facevano parte del gruppo iniziale (Figura 6.9).

Dopo la costruzione Terminata la costruzione si sono verificati ancora alcuni fenomeni interessanti, come ad esempio quello della auto-organizzazione all'interno della comunità. Se per tutta la fase di sviluppo e realizzazione i promotori hanno partecipato attivamente all'attività, svolgendo anche funzioni di coordinamento e gestione della comunità, a cantiere ultimato, immediatamente dopo la rimozione delle centine, i promotori, per motivi legati alla loro attività lavorativa, hanno dovuto smettere o quantomeno diminuire la loro presenza all'interno della discussione. Tuttavia ciò non ha significato la fine dell'esperienza: il resto della comunità si è infatti auto-organizzata per effettuare tutte le rifiniture necessarie sulla volta e per concludere il cantiere. Attraverso l'uso degli strumenti utilizzati in precedenza (principalmente il network sociale) sono stati organizzati degli ulteriori momenti di lavoro che hanno permesso di rifinire tutti i giunti della muratura nell'intradosso della volta (che si presentavano ancora grezzi) e di ripulire e mettere in si-

Figura 6.8: Il passaggio da bit ad atomo presso il laboratorio LATEC: l'immagine proiettata a scala reale viene trasferita manualmente sui supporti di cartoni. Verrà successivamente ritagliata e opportunamente montata.

curezza l'area di cantiere (Figura 6.10). Inoltre la comunità ha continuato la produzione di materiale esplicativo (è stato prodotto un video illustrativo dell'intero processo) e organizzato un momento di inaugurazione in cui è stata invitata tutta la comunità studentesca del Politecnico (Figura 6.11). Oltre a ciò si è cercato di continuare il lavoro sul modello parametrico, attraverso lo sviluppo di alcuni sistemi di controllo di ogni singolo mattone sull'intera superficie di BrickShell.

6.2 Utilità della sperimentazione

6.2.1 BrickShell: Architettura Open Source

Riprendendo alcune delle definizioni presentate nel capitolo precedente si può meglio comprendere perchè BrickShell può essere considerata un'esperienza di Architettura Open Source. La prima definizione, ancora molto generale,

Figura 6.9: Le fasi di cantiere di BrikShell. Vengono posati mattoni seguendo la forma delle centine sottostanti in cartone.

conteneva le condizioni basilari per le quali è possibile parlare o meno di Architettura Open Source:

si parla di Architettura Open Source quando una iniziativa progettuale cerca di adottare il modello Open Source all'interno del suo processo di sviluppo e, per fare ciò, vengono messe in atto adeguate azioni per implementarne la sorgente progettuale e, di conseguenza, creare la comunità che sviluppa tale sorgente e la piattaforma attraverso la quale viene distribuita.

Questa condizione è stata soddisfatta: BrickShell infatti ha attinto a piene mani dalle esperienze di sviluppo open source del software e ha avuto come obiettivi principali quelli di sviluppare una sorgente progettuale adeguata e di costruire una comunità che la sviluppasse attraverso l'uso di una piattaforma dedicata.

A questo punto è forse necessario riflettere circa il motivo per cui BrickShell è diventata un'esperienza di Architettura Open Source e non ha seguito il normale decorso di un workshop tradizionale. Alla base del corso vi era la necessità di affrontare molte tematiche in relativamente poco tempo, e di

Figura 6.10: Brickshell: il padiglione di meditazione del Politecnico di Torino ultimato, poco prima di essere inaugurato.

ottenere un risultato che potesse essere costruito, ovvero un risultato completo di tutte quelle riflessioni che portano un progetto dal foglio da disegno al cantiere. Difficilmente si poteva pensare che gli sforzi di singoli studenti (o gruppi di studenti) potessero riuscire a raggiungere un tale obiettivo. Perciò si è pensato che l'unione degli sforzi in un unico progetto potesse risolvere in parte il problema, riuscendo a portare a termine l'iniziativa e a garantire il completamento dell'edificio. L'open source si sposava in pieno con l'intenzione di gestire un progetto architettonico sviluppato in maniera collaborativa.

L'architettura Open Source non è quindi un fine da perseguire, ma un potente strumento in grado di dare la possibilità a chiunque di sviluppare progetti di architettura e di riplasmare il processo edilizio secondo le proprie necessità.

Si è inoltre ragionato anche sulla effettiva possibilità di realizzazione del prodotto finale. Per fare ciò si è coinvolta da subito l'istituzione ospitante, offrendo un edificio di utilità pubblica e chiedendo in cambio un pezzo di terreno e le necessarie autorizzazioni per procedere alla realizzazione, senza

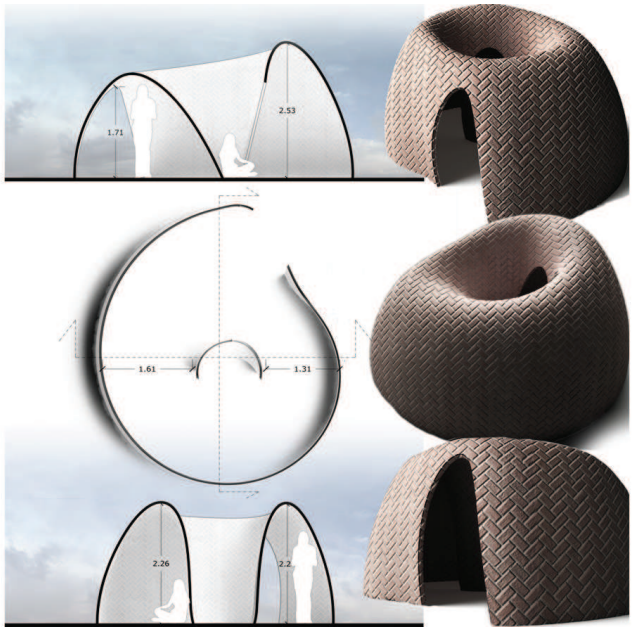

Figura 6.11: La rappresentazione finale del padiglione di meditazione BrickShell.

che vi fosse ancora un progetto definitivo ma solo una funzione prestabilita e alcuni riferimenti. Il coinvolgimento dell'istituzione all'interno del processo ha permesso di ottenere il necessario appoggio per la realizzazione sottolineando quanto non vi sia solo un passaggio da bit ad atomo, ma anche un passaggio da un processo 'informale' ad un contesto normativo consolidato:

Il progetto di Architettura Open Source non deve proporre unicamente un metodo innovativo e accessibile di passaggio da informazione digitale a materia fisica, da bit ad atomo, ma deve anche sforzarsi di sviluppare sistemi adeguati di passaggio da un processo creativo informale ad un contesto costruttivo anche fortemente normato.

6.2.2 BrickShell: sorgente progettuale

Per quanto riguarda la definizione della sorgente, all'interno di BrickShell si è scelto che la sorgente fosse un modello parametrico. Si è visto come questo tipo di strumento permetta due livelli di interazione: un primo livello, che potremmo definire 'alto', consiste nella definizione stessa del modello; mentre un secondo livello, che potremmo definire 'basso', permette, attraverso la variazione dei parametri, di modificare non la definizione del modello ma la sua forma finale adattandola alle proprie necessità. Questo tipo di interazione è piuttosto inusuale poiché normalmente nelle altre esperienze che

sono state affrontate non esiste questa differenza, ma esiste unicamente un livello 'alto'. Per poter partecipare allo sviluppo di un software bisogna saper programmare, per un progetto di Open Design bisogna saper modellare e in generale bisogna saper svolgere delle attività che l'utente medio difficilmente è in grado di svolgere. Con il modello parametrico si riesce invece ad attuare un sistema di risultato immediato con relativa facilità, che può spingere l'utente a comprendere meglio l'essenza della sorgente e a dotarsi degli strumenti necessari per operarvi al meglio e passare dunque da un livello 'alto' a un livello 'basso':

una iniziativa di architettura open source può dirsi tale non soltanto se la sorgente è distribuita liberamente, ma se essa, per sua stessa natura, favorisce la modifica e l'implementazione da parte di altri utenti, anche non esperti.

Per implementare ulteriormente la fruizione della sorgente e la sua effettiva apertura si è detto come il ricorso a sistemi modulari e dimensioni standard possa avere effetti positivi. Nel caso BrickShell non solo si è fatto ricorso a elementi modulari (come ad esempio le centine), ma l'intero progetto delle opere provvisionali si è basato sulla dimensione del mattone (nel caso specifico sono stati utilizzati mattoni 4,5 x 15 x 30 cm). Inoltre il modello parametrico ha permesso che questa dimensione fosse variabile. L'utilizzo di moduli ricorrenti ha quindi permesso di realizzare con facilità un oggetto dalla forma complessa.

La presenza di dimensioni standard e moduli ricorrenti diventa un efficace strumento progettuale, che, al contrario di quanto si potrebbe pensare, non favorisce la ripetitività ma esalta le differenze tra le varie soluzioni sviluppate in seguito all'attività di ricombinazione delle parti.

In ultimo conviene affrontare la questione relativa al passaggio dal bit all'atomo. Se per l'Open Design questo viene fatto attraverso macchine a controllo numerico, che permettono il controllo di tolleranze minime e la produzione anche molto limitata di pezzi senza costi particolari, per l'Architettura Open Source si sono visti diversi approcci. WikiHouse abbraccia in toto la fabbricazione digitale; mentre esperienze come Open Source Ecology utilizzano sistemi molto tradizionali come la muratura.

In BrickShell si è visto come il passaggio da informazione digitale a materia fisica possa avvenire seguendo tecniche miste in mancanza di un supporto economico adeguato. Infatti ciò non ha impedito la corretta esecuzione dell'opera poiché la centina realizzata ha permesso di realizzare una forma analoga a quella presente nel modello virtuale.

Nel passaggio da bit ad atomo non è dunque obbligatoria la fabbricazione digitale, tuttavia devono sempre essere previsti adeguati

strumenti (tendenzialmente digitali) volti a facilitare il passaggio da bit a atomo, riducendo la possibilità di errore in fase di esecuzione.

6.2.3 BrickShell: comunità di pari

> *"The notion of shared authorship had important implications for the design process. As Wojtowicz (1994) observed in a somewhat similar VDS experiment conducted in 1993, «designer privacy is breached [...] a designer has to give up the privacy protecting his or her own design process and at the same time is exposed to a surrounding context [...] which is constantly modified by other members.» Surprisingly, in our experiment, the fact that no individual ownership of a design is possible seems not to pose a problem to anyone. Perhaps this is due to the difference in nature between the university environment and professional practice, and that the designs were abstract and of short duration. Yet the people we asked could imagine working in practice under similar conditions. Therefore, such an approach might be a strong hint to a possible future AEC working environment. The premise was that the development of a new design and collaboration environment, along with a new collaboration method, could result in a breakthrough of productivity and quality."*[17]
>
> (Schmitt et al. 1999b)

In riferimento alla gestione della comunità, BrickShell ha fornito alcune interessanti indicazioni. Innanzitutto è conveniente riprendere una delle affermazioni del capitolo precedente:

[17]"La nozione di autorialità condivisa ha avuto importanti implicazione per il processo progettuale. Come Wojtowicz (1994) ha osservato in una esperienza simile condotta nel 1993 «la privacy del progettista è violata [...] un progettista deve abbandonare la privacy che gli permette di proteggere il suo processo progettuale e al contempo si espone a un contesto circostante che viene continuamente modificato dagli altri membri». Sorprendentemente, nel nostro esperimento, il fatto che non fosse possibile apporre proprietà intellettuali singole sui progetti non ha posto problemi a nessuno. Probabilmente ciò è dovuto alla differenza che intercorre tra la pratica professionale e l'ambiente universitario, e che i progetti erano astratti e lo sviluppo di breve durata. Eppure le persone a cui abbiamo domandato hanno risposto che lavorerebbero in simili condizioni. Pertanto un tale approccio potrebbe essere un forte indizio per un possibile sviluppo futuro dell'ambiente di lavoro AEC. La premessa era che lo sviluppo di un nuovo ambiente di progettazione, insieme a un nuovo metodo di collaborazione, potesse provocare una svolta di produttività e qualità." [traduzione italiana a cura dell'autore].

Si può affermare che la comunità è composta da tutti gli utenti che partecipano a una iniziativa di architettura open source in maniera attiva o indiretta.

Per BrickShell è avvenuto effettivamente così: moltissimo è stato svolto da utenti sviluppatori, ma molto è stato fatto anche da tutte le altre persone coinvolte che non hanno operato direttamente sulla sorgente, ma che attraverso altre operazioni, di tipo indiretto, hanno permesso che l'intera iniziativa potesse avere luogo. Il progetto di Architettura Open Source, piuttosto che a ridurre il numero di persone coinvolte, tende infatti ad allargarlo. Questo per un semplice motivo: a ognuno viene chiesto un minimo sforzo e la totalità dei minimi sforzi, se adeguatamente organizzati e sfruttati, garantirà una massa critica sufficiente a permettere lo sviluppo del progetto. Per fare un esempio banale si può prendere in considerazione la comunità iniziale, composta da 5 docenti (utenti promotori) e 16 studenti (utenti sviluppatori). Il rapporto è all'incirca di uno a tre. Se consideriamo il fatto che si è lavorato su di un unico progetto, il rapporto è di un progetto con ventuno progettisti. Vedendo l'entità del progetto, il numero di progettisti è chiaramente spropositato ma il sistema derivato dell'Open Source permette di organizzare tanti piccoli sforzi in maniera efficace. Ciò è stato ancora più evidente nella fase di costruzione: la muratura prevede infatti che si posi un mattone dopo l'altro senza necessarie capacità, la cosa importante è seguire il pattern scelto (in questo caso non a correre ma a spina di pesce). Durante la costruzione sono state coinvolte almeno altre dieci persone oltre alla comunità iniziale, coinvolgimenti di varia entità come colleghi, amici e parenti. Sono stati posati circa un migliaio di mattoni, il che significa che in media ogni utente ha posato circa trenta mattoni. Trenta mattoni è una cifra irrisoria, ma ogni singolo minimo sforzo, inquadrato in un sistema capace di valorizzarlo e di massimizzarne l'impatto, ha avuto un effetto globale di un certo rilievo. Probabilmente due muratori esperti avrebbero terminato il lavoro in meno tempo e il risultato finale sarebbe stato grosso modo lo stesso.

Un altro aspetto che merita di essere affrontato è quello legato all'autorialità, la quale potrebbe essere un problema soprattutto se la comunità dovesse assorbire il lavoro dei singoli senza riconoscerne gli sforzi. Ma nella pratica succede l'esatto contrario, giacché solo attraverso il riconoscimento del lavoro del singolo la comunità può continuare il lavoro di sviluppo. Se teniamo conto che ogni singola azione viene registrata sulla piattaforma, il problema della autorialità non sussiste: ogni modifica, ogni apporto, viene registrato sulla piattaforma.

Il lavoro di ciascun utente viene registrato all'interno della piattaforma e viene riconosciuto all'interno della comunità, la quale ne valuterà la portata e ne garantirà la corretta distribuzione.

Una degli aspetti meno sviluppati in BrickShell è stato quello relativo alla organizzazione modulare della comunità. Ciò non è stato fatto poiché si trattava di una comunità già piuttosto piccola e avrebbe avuto poco senso frammentarla ulteriormente. Ma nonostante ciò si è verificato comunque un fenomeno interessante: nel momento in cui è mancata la partecipazione attiva dei promotori, l'attività non è cessata ma è continuata secondo gli obiettivi posti in precedenza. Ciò significa che non è tanto importante stabilire una divisione delle competenze a priori quanto piuttosto evidenziare efficacemente gli obiettivi da raggiungere: in base a questi la comunità si auto-organizzerà efficacemente senza la necessità di una guida in particolare.

6.2.4 BrickShell: piattaforme di supporto

La piattaforma è l'insieme di strumenti digitali che permettono alla comunità di distribuire, implementare e sviluppare la sorgente.

Si è visto come la definizione della piattaforma di BrickShell si sia basata su strumenti già esistenti che, composti e ricombinati tra di loro, hanno permesso di formare un unico strumento in grado di ospitare e garantire il lavoro della comunità. Utilizzare strumenti già esistenti significa che non è tanto la definizione degli strumenti in quanto tali che caratterizza in positivo o in negativo la piattaforma, quanto piuttosto la definizione degli usi che se ne deve fare. Infatti lo svolgimento di esercizi propedeutici, il ricorso alla piattaforma anche nella fase iniziale e più tradizionale del workshop sono elementi che concorrono a un uso corretto della piattaforma da parte degli utenti. Ma la vera differenza la fanno le scelte che stabiliscono le modalità di lavoro collaborativo: scegliere di non seguire il modello 'staffetta' ma abbracciare in pieno il modello Open Source significa correre il rischio di piombare nel caos, ma vuol dire anche spostare l'attenzione dallo strumento in sé (sito web, network sociale) alle modalità di uso creativo e collaborativo che è possibile innescare.

La piattaforma è oggetto, al pari della sorgente, di progetto da parte dei promotori e per esteso della comunità.

La scelta di utilizzare tutti strumenti liberamente scaricabili e utilizzabili (anche se molto software utilizzato non è open source) favorisce ovviamente il grado di apertura della sorgente. Chiunque è in grado di scaricare il software, installarlo sul proprio computer e utilizzarlo. Inoltre la scelta di utilizzare il software parametrico permette diversi livelli di interazione, stimolando la curiosità degli utenti più esperti ma favorendo comunque l'accesso di utenti meno esperti.

Non vi è dunque un livello minimo di interazione da raggiungere, ma più alta sarà l'interattività della piattaforma più accessibile e aperta risulterà la sorgente.

6.2.5 BrickShell: considerazioni generali e risultati

Il workshop BrickShell è stato il banco di prova della quasi totalità degli strumenti operativi dell'Architettura Open Source. Ne ha dimostrato il funzionamento in un contesto ristretto e in un periodo di tempo limitato. Attualmente è in fase di programmazione una seconda esperienza BrickShell, i cui obiettivi sono quelli di strutturare il processo collaborativo in modo che esso possa essere ripetuto anche in altre circostanze e, dal punto di vista pratico, di minimizzare i tempi di preparazione del cantiere e di costruzione.

Quello che emerge da questa esperienza è che lo strumento dell'Architettura Open Source, se attentamente progettato, può essere applicato con successo. Ma come bisogna agire per utilizzare correttamente uno strumento come l'Architettura Open Source?

Quando Stallman definì il free software e la licenza copyleft che ne permetteva l'esistenza, l'Open Source non esisteva: mancava un processo definito ed esplicito che ne permettesse l'espansione. L'apporto di Linus Torvalds (poi efficacemente sintetizzato da Raymond) è stato fondamentale: da un lato si è trattato di una richiesta assai banale di aiuto per effettuare dei test di affidabilità del sistema, dall'altro si è immediatamente trasformato in una presa di coscienza degli appassionati che da amici e colleghi di Linus si sono trasformati nella prima comunità open source, stabilendo le regole di un processo che ha fatto scuola. Linus ha saputo rendere esplicite delle regole che prima di allora erano solo implicite, regole legate alla tradizione hacker e dei primi programmatori. Tali regole sono state rese esplicite insieme con gli strumenti attraverso cui metterle in atto, ad esempio sviluppando software come Git[18], un programma che rende possibile il lavoro di più sviluppatori software su di una sola sorgente, attraverso delle regole definite. Proprio sulla base di quelle regole si sono sviluppati migliaia di progetti di software open source. Wikipedia ha fatto lo stesso attraverso lo strumento del Wiki e le regole di redazione, sviluppando milioni di pagine della nota enciclopedia libera.

L'Architettura Open Source dovrebbe seguire lo stesso modello. La redazione degli strumenti pratici e la loro messa in opera attraverso BrickShell ne sono un tentativo. L'Open Source è uno strumento di sviluppo che prevede la presenza di una sorgente aperta. Ma il metodo di sviluppo stesso è un metodo aperto che può essere utilizzato da chiunque.

[18] `http://git-scm.com/`

Gli strumenti operativi per l'Architettura Open Source possono essere essi stessi considerati una sorgente, dai quali è possibile sviluppare numerosi progetti, anche molto diversi tra loro. BrickShell ne ha dimostrato il funzionamento e sarà compito degli architetti, se lo vorranno e lo riterranno opportuno, utilizzarli per innescare processi di trasformazione dell'ambiente costruito collaborativi e inclusivi.

Conclusioni e sviluppi futuri

L'Architettura Open Source è un fenomeno recente, in continua evoluzione. Si riferisce a una tradizione consolidata legata all'idea di partecipazione e democratizzazione del processo edilizio e, attraverso strumenti e tecniche proprie, realizza dei modelli di gestione e sviluppo del processo progettuale con caratteristiche specifiche. Di questo fenomeno, seppur ancora in evoluzione, è stato possibile dare una serie di definizioni programmatiche basandosi principalmente sulla lettura dei casi studio emergenti. É stato possibile riconoscere all'Architettura Open Source alcune potenzialità ed effetti, alcune dei quali non si sono ancora manifestati nella loro interezza, e che è quindi possibile solo ipotizzare. Per quanto riguarda le potenzialità e gli effetti già manifestatisi, è bene ricordarne almeno due: la prima è relativa alla transcalarità, ovvero la capacità dello strumento open source di funzionare su varie scale, da quella locale a quella globale; la seconda è legata al ruolo che il professionista è chiamato a ricoprire in un processo aperto.

Transcalarità

Sono state prese in esame delle iniziative sperimentali che, proprio perché sperimentali, presentano anche dei limiti. Nonostante questo vi sono almeno due progetti, WikiHouse e Open Source Ecology, che sembrano aver superato la fase di sviluppo iniziale e sono riuscite a consolidarsi positivamente. È indubbio infatti che tutte le esperienze viste siano state salutate all'inizio della loro attività con un certo entusiasmo e positivo ottimismo. Tuttavia, alcune si sono dovute scontrare con dei limiti strutturali interni che ne hanno circoscritto fortemente l'attività, in certi casi fino alla sparizione. Nel caso delle due iniziative citate sopra le cose sono andate diversamente, sia per la portata e la qualità della proposta, sia per la capacità di espandersi al di fuori dei confini nazionali raggiungendo un livello di espansione globale. Infatti attraverso la creazione di sotto-comunità locali si è assistito a un vero e proprio salto di scala che fino ad allora era stato immaginato da molti (nel caso di OpenSimSim con l'iniziativa OpenJapan era stato fatto anche un tentativo di messa in atto) ma non era stato ancora consolidato.

Se da un lato abbiamo la conferma che è possibile costruire e realizzare progetti a impatto globale, dall'altro l'esperienza di Brickshell conferma che anche su piccola scala l'Open Source può essere uno strumento molto efficace. Oltre a questa considerazione si può aggiungere che la varietà dei temi affrontati (sviluppo rurale, abitazioni peri-urbane, didattica e formazione) sono una prova dell'efficacia dell'Architettura Open Source e della sua versatilità. L'architettura Open Source ha sviluppato la capacità di superare la soglia del locale riuscendo ad avere un impatto globale, senza che questo ne limiti l'applicazione anche in contesti differenti. Possiamo quindi dire

che l'Architettura Open Source è 'glocal': in grado di gestire globalmente le istanze locali e di adattarsi a specifici contesti attraverso il lavoro di comunità transcalari di utenti.

Un nuovo ruolo per l'architetto?

Un altro aspetto interessante è legato al ruolo che il progettista si ritaglia in attività di tipo 'open'. Si è già detto come venga affrontato il problema autoriale: l'autore non è privato del suo diritto autoriale, ma entra in un sistema differente dove l'autorialità assume connotati diversi da quelli ordinari. Se la figura del promotore ricalca quella di un manager e di una guida, nel momento in cui si è sviluppatori sembra quasi di essere al servizio delle comunità senza che questo servizio venga remunerato. Nella realtà, proprio perché la comunità si auto-organizza, ognuno è chiamato a ritagliarsi un ruolo secondo le sue personali abilità o attitudini.

Nel corso degli ultimi anni il lavoro dell'architetto ha visto emergere numerose specializzazioni (legate alla sicurezza, al risparmi energetico, alla rappresentazione etc.). Nelle comunità online accade lo stesso fenomeno: ogni utente acquisisce una serie di competenze e di conoscenze specifiche e la sua attività all'interno della comunità finisce per acquisire una particolare connotazione. Semplicemente, non esiste più un ruolo definito a priori (attraverso l'acquisizione di un titolo professionale, ad esempio) mentre appare determinante il lavoro svolto all'interno della comunità, governato dai principi di auto-organizzazione e di reputazione, e ciò contribuisce a definire nuovi ruoli e mandati. L'utente è ciò che fa, ciò che impara a fare, ciò che spiega e insegna agli altri; la comunità diviene al contempo ente giudicante e struttura di sostegno. Sarà compito preciso degli architetti quello di costruire un ruolo che non ne sminuisca l'attività ma che diventi proficuo e necessario per la collettività.

Il dibattito continua

Nel momento in cui questo lavoro è in fase di completamento il dibattito sull'Architettura Open Source continua. Nel novembre 2013 è stato lanciato da Architecture for Humanity di Denver, insieme con Open Tech Forever (un collettivo di costruttori e agricoltori precedentemente affiliati a Open Source Ecology) un concorso[19] aperto per la progettazione di un modulo abitativo in blocchi di terra compressa (da costruirsi con la CEB press di OSE) per lo sviluppo di una comunità agricola nei pressi di Denver. Il titolo del concorso è

[19] `https://github.com/AfH-Denver/Forever_Home_Design_Challenge/wiki`

Forever Home: Open Source Building Design Challenge. La richiesta è quella di avere il concept di una unità abitativa con possibilità di facili modifche da parte dell'utente, e che rispetti degli standard costruttivi anche piuttosto elevati in termini di consumo energetico e sostenibilità ambientale[20]. Il concorso è terminato il 21 gennaio 2014 e ad oggi ancora non è possibile sapere i vincitori né tantomeno l'uso che verrà fatto dell'intero corpus di progetti presentati. Ciò che è chiaro però è che, seppur in via sperimentale, l'Architettura Open Source offre dei validi strumenti per favorire l'incontro tra una committenza organizzata (una comunità vera e propria, non online) e il possibile progettista. La forma del concorso (basato sulla competitività) stride con l'idea della collaborazione, ma ciò non vuol dire che a partire da una buona idea progettuale non sia possibile in seguito sviluppare una sorgente valida. L'idea è proprio questa: ottenere un progetto flessibile che possa essere sviluppato dalla comunità agricola di Denver per costruire tutte le abitazioni necessarie sui 40 acri di terreno in Colorado.

Verso la fine di novembre 2013, sul website Archdaily[21] (una sorta di rivista di architettura online) è stato pubblicato un articolo dal titolo *Paperhouses: Architecture in Open Source*[22], scritto da Vanessa Quirk. L'articolo illustra il progetto Paperhouses[23], il quale si pone l'obiettivo di rendere disponibili interi progetti di abitazioni unifamiliari, progetti già sviluppati in precedenza da vari studi di architettura che partecipano all'iniziativa insieme ai promotori. "The idea behind Paperhouses, founded by Joana Pacheco, is one based on collaboration: world-class architects freely provide world-class architecture to professionals and laymen, who can then download and adapt these plans as they see fit (adjusting the design for program, square footage, climate, etc.) In this act, a conversation – between clients, builders, and architects – is born."[24](Quirk 2013). Si parte non da una necessità (quella del cliente) ma da una possibile risposta (la casa disponibile su Paperhouses) per innescare un nuovo processo edilizio. Paperhouses funge così da repertorio pronto all'uso che ognuno degli attori può usare a proprio piacimento. Al momento si può accedere unicamente alle descrizioni del progetto ma le prime piante non sono ancora disponibili, per questo motivo non è stata inclusa

[20] `http://living-future.org/lbc/about`

[21] `http://www.archdaily.com/`

[22] `http://www.archdaily.com/452176/paperhouses-architecture-in-open-source/`

[23] `http://www.paperhouses.co/`

[24] "L'idea che sta dietro a Paperhouses, fondata da Joana Pacheco, è basata sulla collaborazione: architetti rinomati rendono disponibili gratuitamente alcune loro architetture ad altri professionisti e addetti ai lavori, i quali possono scaricare le piante e adattarle alle proprie necessità (dal punto di vista funzionale, delle dimensioni, del clima). Attraverso questo processo si sviluppa un nuovo livello di dialogo tra costruttori, progettisti e clienti." [traduzione italiana a cura dell'autore].

nei casi studio. Rimane comunque difficile pensare che una iniziativa di questo tipo possa essere considerata effettivamente Architettura Open Source alla luce di quanto visto finora, ma la sola esistenza è sintomatica del fatto che il dibattito sull'Architettura Open Source continua, e che gli architetti vi vedono una possibilità, anche economica, di affrontare gli sviluppi della pratica professionale nel mondo contemporaneo.

Open Source e pratica professionale

Oltre al salto di scala, che sta già avvenendo grazie ad alcune iniziative citate, l'Architettura Open Source potrà consolidarsi se riuscirà a uscire dal limbo dell'informalità per approdare a uno status di pratica comune normativamente riconosciuta. Esattamente come il salto di scala, tale passaggio avverrà grazie alla possibilità di trasferire localmente le potenzialità e le risorse cognitive espresse a livello globale. Ciò significa che saranno necessari dei traduttori in grado di trasferire in contesti specifici istanze generali e globali. Questi saranno gli architetti, che avranno il compito, se lo vorranno e se lo riterranno giusto, di utilizzare l'Open Source come valido strumento di gestione del processo edilizio. Non sarà forse possibile costruire dei grattacieli open source, ma già oggi la complessità di alcuni interventi fa pensare che un approccio di tipo tradizionale non possa più funzionare. Se effetivamente "gli architetti dovranno verosimilmente confrontarsi con una maggiore domanda sociale, e con una diminuzione nella domanda di prodotti di lusso" (Carpo 2009b, p. 22) sarà bene che si dotino degli strumenti necessari.

In un sistema in cui l'utilizzo di macchinari per la fabbricazione digitale sarà sempre più accessibile e conveniente, e in cui gli strumenti e le interfacce di connessione e di collaborazione tra utenti diversi diventeranno via via sempre più efficienti, performanti e accessibili, l'Architettura Open Source sarà un valido strumento solo se i professionisti decideranno di adottarlo. Sarà necessario dare più importanza al processo rispetto al progetto finale, sarà necessario capire come trasmettere ai futuri utenti e alle amministrazioni il valore di un processo inclusivo rispetto a quello di un progetto esclusivo. Senza l'apporto dei professionisti sarà impensabile anche solo prevedere uno sviluppo futuro che si differenzi dalla situazione attuale: senza cambio di passo si continuerà a proporre iniziative di Architettura Open Source che non saranno in grado di incidere sulle trasformazioni dell'ambiente costruito, ma continueranno a rimanere in uno status di 'esperimento continuo'. Per nostra fortuna in paesi come gli Stati Uniti e il Regno Unito questo passaggio sarà forse più facile, infatti non a caso Wikihouse e Open Source Ecology si stanno sviluppando con una certa fortuna proprio in quei paesi. Sarà necessario

mettere a punto degli strumenti ad hoc per fare in modo che la spinta globale di queste iniziative non si esaurisca di fronte a ostacoli burocratici locali.

Conclusioni

L'Architettura Open Source sta diventando una realtà nel panorama architettonico contemporaneo. Dall'inizio della rivoluzione digitale ad oggi la pratica architettonica ha cercato di sfruttare appieno le potenzialità che i nuovi strumenti informatici sono in grado di fornire. Ora sta iniziando a sfruttarne anche le potenzialità connettive oltre a quelle computative, e tale operazione sta lentamente portando il processo architettonico da un processo chiuso quale era, popolato da poche e chiare presenze (l'architetto, il cliente e l'impresa) a un processo aperto, collettivo, plurale, in cui ruoli e mandati sono in continua fase di ridefinizione e riscrittura.

L'Open Source offre un modello di organizzazione del lavoro profondamente diverso da quanto eravamo abituati a vedere, radicalmente alternativo alla pratica tradizionale ma non per questo in assoluta antitesi. È uno strumento potente che, se usato a dovere, può permettere di affrontare con relativa facilità situazioni complesse.

Alla luce della crisi economica che stiamo vivendo, della sempre più palese scarsità di risorse, della richiesta di democratizzazione dei processi di trasformazione, l'Open Source sembra in grado di offrire una modalità inaspettatamente efficace di uso dei nuovi media e degli strumenti digitali. È necessario rendersi conto che in molti altri settori l'Open Source è stato adottato positivamente per affrontare i problemi che la complessità dei rapporti contemporanei ha posto in essere. Se anche l'Architettura saprà farlo, avrà uno strumento in più per affrontare con efficacia le sfide progettuali che il nuovo millennio porta con sè.

Bibliografia

Abbate, J. (1999). *Inventing the Internet*, MIT Press.

Alexander, C. (1964). *Notes on the Synthesis of Form*, Harvard University Press.

Alexander, C. (1973). *Notes on the Synthesis of Form*, Harvard University Press.

Alexander, C. (1979). *The timeless way of building*, Oxford University Press.

Alexander, C., Davis, H., Martinez, J. & Corner, D. (1985). *The Production of Houses*, Oxford University Press.

Alexander, C., Ishikawa, S., Silverstein, M., Jacobson, M., Fiksdahl-King, I. & Angel, S. (1977). *A Pattern Language*, Oxford University Press.

Anderson, C. (2010a). In the next industrial revolution, atoms are the new bits, *Wired* **18**.

Anderson, C. (2010b). *La coda lunga*, Codice Edizioni.

Arthur, W. B. (2011). *La natura della tecnologia*, Codice Edizioni.

Avital, M. (2011). The generative bedrock of open design, *in* B. van Abel, R. Klaasen, L. Evers & P. Troxler (eds), *Open design now*, BIS.

Benkler, Y. (2007). *La ricchezza della rete*, Universitá Bocconi Editore.

Berners-Lee, T. (1999). *Weavin the Web*, Harper.

Berra, M. & Meo, A. R. (2001). *Informatica solidale: storia e prospettive del software libero*, Bollati Boringhieri.

Bocco, A. & Cavagliá, G. (2008). *Cultura tecnologica dell'architettura*, Carocci.

Boudon, P. (1983). *Pessac di Le Corbusier*, Franco Angeli.

Bourriaud, N. (2002). *Postproduction, Culture as Screenplay: How Art Reprograms The World*, Lukas & Sternberg.

Cache, B. (2011). *Projectiles*, Architectural Association London.

Calvino, I. (1993). *Lezioni americane. Sei proposte per il prossimo millennio*, Mondadori.

Carpo, M. (2009a). Authors, agents, agencies, and the digital public, *in* M. Brizzi & P. Giaconia (eds), *Visions*, Image Publishing.

Carpo, M. (2009b). The bubble and the blob, *Lotus International* **138**: 19–26.

Carpo, M. (2011). *The alphabet and the algorithm*, The MIT Press.

Castells, M. (2001). L'informazionalismo e la network society, *in* P. Himanen (ed.), *L'etica hacker*, Feltrinelli.

Castells, M. (2002). *Galassia Internet*, Feltrinelli.

Ceragioli, G. (2002). *Dare unanima al futuro: note per un umanesimo tecnologico*, Mille.

Dawkins, R. (1995). *Il gene egoista*, Mondadori.

DeLanda, M. (2001). Open-source: A movement in search of a philosophy, Institute for Advanced Study, Princeton, New Jersey.

Eco, U. (1958). Il problema dell'opera aperta, *XII Congresso Internazionale di Filosofia, Venezia.*

Eco, U. (1962). *Opera aperta*, Bompiani.

Fabris, L. M. F. (2002). *Metodo Segal*, Libreria Clup.

Floridi, L. (2012). *La rivoluzione dell'informazione*, Codice Edizioni.

Foti, M. (2005). *Tecnologie per lo sviluppo : note del gruppo Ceragioli per una progettazione etica*, Politecnico di Torino.

Frazer, J. (1982). The use of simplified three-dimensional computer input devices to encourage public participation in design, *in Computer-Aided Design Conference proceedings, 03/1982.*

Friedman, Y. (1972). *Per una architettura scientifica*, Officina Edizioni.

Friedman, Y. (1975). Computer - aided parcipatory design, *in* N. Negroponte (ed.), *Soft Architecture Machines*, The MIT Press.

Friedman, Y. (2000). *Utopie realizzabili*, Quodlibet.

Friedman, Y. (2009). *Larchitettura di sopravvivenza: una filosofia della povertá*, Bollati Boringhieri.

Gallanti, F. (2005). Elemental, aravena!, *Domus* **886**: 34 – 41.

Gershenfeld, N. (2005). *Fab - Dal personal computer al personal fabricator*, Codice Edizioni.

Goodman, N. (1976). *I linguaggi dell'arte*, Il Saggiatore.

Gubitosa, C. (1999). La vera,storia di internet, `http://www.apogeonline.com/2001/libri/88-503-1055-2/ebook/pdf/StoriaInternet.pdf`, ultima visita novembre 2013.

Habraken, N. J. (1974). *Strutture per una residenza alternativa*, il Saggiatore.

Habraken, N. J. (n.d.). Molenvliet project, `http://www.habraken.com/html/molenvliet.htm`, ultima visita novembre 2013.

Haque, U. (2002). Hardspace, softspace and the possibilities of open source architecture, `http://www.haque.co.uk/papers.php`, ultima visita novembre 2013.

Hertzberger, H. (1991). *Lessons for Students in Architecture*, 001 Publishers.

Himanen, P. (2001). *L'etica Hacker*, Feltrinelli.

Illich, I. (1974). *La convivialitá*, Mondadori.

Jenkins, H. (2007). *Cultura convergente*, Apogeo.

Kaspori, D. (2003). A communism of ideas. towards an open-source architectural practice, *Archis* **3**: 13 –17.

Kelly, K. (2011). *Quello che vuole la tecnologia*, Codice Edizioni.

Kendall, S. (2000). Next 21, in cib w104 open building implementation, `http://open-building.org/ob/next21.html`, ultima visita novembre 2013.

Kennedy, J. F. (1994). Steve wozniak: hacker and humanitarian, *in* G. Kawasaki (ed.), *Hindsights-The Wisdom and Breakthroughs of Remarkable People*, Beyond Words.

Kurzweil, R. (2008). *La singolaritá é vicina*, Apogeo.

Kuwabara, K. (2000). Linux: A bazaar at the edge of chaos, *First Monday* **5**.

Larson, K., Intille, S., McLeish, T., Beaudin, J. & Williams, R. (2004). Open source building reinventing places of living, *BT Technology Journal* **Vol 22 No 4**: 187–200.

Le Corbusier, . (1953). *Le Corbusier & P. Jeanneret : oeuvre compléte, 1934-1938*, Girsberger.

Lessig, L. (2006). *Il futuro delle idee*, Feltrinelli.

Lessig, L. (2009). *Remix: il futuro del copyright (e delle nuove generazioni)*, ETAS.

Levy, P. (1996a). *L'intelligenza collettiva: per un'antropologia del cyberspazio*, Feltrinelli.

Levy, S. (1996b). *Hacker - Gli eroi della rivoluzione informatica*, ShaKe Edizioni.

Lommée, T. (2009). Introduction, in openstructures, febbraio 1997, `http://openstructures.net/`, ultima visita gennaio 2014.

Lommée, T. (2011). Un esperanto per gli oggetti, *Domus* **648**: 88–95.

Manaugh, G. (2011). Verso un'ecologia open source, *Domus* **948**: 82–87.

Marchetti, L. (2006). *Arte ed estetica in Nelson Goodman*, Centro Internazionale Studi di Estetica.

McKean, J. (1986). Semi preziosi di buon senso, *Spazio e Societá* **34**: 18–26.

Menichinelli, M. (2008). Openp2pdesign 1.1 - design for complexity, in `http://www.openp2pdesign.org/source/?download=6`, ultima visita novembre 2013.

Moglen, E. (1999). Anarchism triumphant: Free software and the death of copyright, *First Monday* **4**.

Negroponte, N. (1974). *La macchina per l'architettura*, il Saggiatore.

Negroponte, N. (1975). *Soft Architecture Machines*, The MIT Press.

Negroponte, N. (1995). *Essere digitali*, Sperling & Kupfer.

Parvin, A., Saxby, D., Cerulli, C. & Schneider, T. (2011). *A right to build, The next mass-housebuilding industry*, University of Sheffield School of Architecture and Architecture 00:/.

Pathiraja, M. (2010). Procurement strategy as means of capacity building in sri lanka: Architecture, design and labour training, *Proceedings: W092 - Special Track 18th CIB World Building Congress*.

Picon, A. (2010). *Digital culture in architecture: an introduction for the design professions*, Birkher.

Potter, N. (2002). *Cos'é un designer*, Codice Edizioni.

Prestinenza Puglisi, L. (2013). Ecologia e rifiuto dei valori urbani, in storia dell'architettura 1905-2008, `http://presstletter.com/2013/01/6-2-6-ecologia-e-rifiuto-dei-valori-urbani/`, ultima visita gennaio 2014.

Pugnale, A. (2007). Terminologia specifica, un mini-glossario, in www.albertopugnale.com, `http://albertopugnale.wordpress.com/2007/03/12/i-termini-di-moda-e-i-termini-specifici-un-mini-glossario/`, ultima visita gennaio 2014.

Quirk, V. (2013). Paperhouses: Architecture in open source, in archdaily `http://www.archdaily.com/452176/paperhouses-architecture-in-open-source/`, ultima visita gennaio 2014.

Ratti, C., Antonelli, P., Bly, A., Dietrich, L., Grima, J., Hill, D., Habraken, J., Haw, A., Maeda, J., Negroponte, N., Obrist, H. U., Reas, C., Santambrogio, M., Somajni, C., Sterling, B. & Shepard, M. (2011a). Editoriale: open source architecture (osarc), *Domus* **948**: i–iv.

Ratti, C., Antonelli, P., Bly, A., Dietrich, L., Grima, J., Hill, D., Habraken, J., Haw, A., Maeda, J., Negroponte, N., Obrist, H. U., Reas, C., Santambrogio, M., Somajni, C., Sterling, B. & Shepard, M. (2011b). Opensource architecture, `http://en.wikipedia.org/wiki/Opensource_Architecture`, ultima visita novembre 2013.

Raymond, E. S. (1998). *La cattedrale e il bazzar*, Apogeonline.

Rouillard, D. (2004). *Superarchitecture, le future de larchitecture 1950-1970*, editions de la Villette.

Salingaros, N. A. (1997). Influence on computer science, in some notes on christopher alexander, `http://www.math.utsa.edu/~yxk833/Chris.text.html`, ultima visita novembre 2013.

Sampó, L. (2013). Open-source culture, *Boundaries* **7**: 4–11.

Schmitt, G., Hirschberg, U., Kurmann, D., Kolarevic, B., Johnson, B. & Donath, D. (1999a). The 24 hour design cycle: an experiment in design collaboration over the internet., *Fourth Conference on Computer-Aided Architectural Design Research in Asia - CAADRIA*, Tonji University, pp. 181–190.

Schmitt, G., Hirschberg, U., Kurmann, D., Kolarevic, B., Johnson, B. & Donath, D. (1999b). An experiment in design collaboration, *ACADIA Conference Proceedings*, University of Cincinnati, pp. 90–99.

Senge, P. (1992). *La quinta disciplina: L'arte e la pratica dell'apprendimento organizzativo*, Sperling & Kupfer.

Silvestri, S. (2009). Favelas e social housing. un codice generativo per l'edilizia sociale, *Bioarchitettura* **56**: 52–59.

Sinclair, C. (2006). Open-source architecture, ted talk, `http://www.ted.com/talks/cameron_sinclair_on_open_source_architecture.html`, ultima visita novembre 2013.

Solazzo, V. (2009). N. john habraken: Architecture and the problem of everyday environment, gennaio 2009, `http://www.arc1.uniroma1.it/saggio/Avvenimenti/NewWorld/VSolazzojohnhabraken.pdf`, ultima visita gennaio 2014.

Stallman, R. (2003). *Software libero, pensiero libero*, Stampa Alternativa.

Sterling, B. (2006). *La forma del futuro*, Apogeo.

Surowiecki, J. (2007). *La saggezza della folla*, Fusi orari.

Sutphen, S., Ehud Sharlin, E., Watson, B. & Frazer, J. (2000). Reviving a tangible interface affording 3d spatial interaction, *In Proceedings of 11th Western Canadian Computer Graphics Symposium*.

Tafuri, M. (2007). *Progetto e utopia*, Laterza.

Tosoni, P. (1999). *Derive della cultura architettonica*, Celid.

Tosoni, P. (2008). *Il gioco paziente*, Celid.

Turin, D. A. (2003). Building as a process, *Building Research & Information* **31(2)**: 180187.

van Abel, B., Klaasen, R. & Evers, L. (2011). Preface, *in* B. van Abel, R. Klaasen, L. Evers & P. Troxler (eds), *Open design now*, BIS.

Vardouli, T. (2011a). The emergence of participatory techno-utopias: Geam, giap and yona friedman, in openarchitecture(s), `http://openarchitectures.wordpress.com/2011/09/19/19/`, ultima visita novembre 2013.

Vardouli, T. (2011b). Prolegomena, in openarchitecture(s), `http://openarchitectures.wordpress.com/2011/09/18/prolegomena/`, ultima visita novembre 2013.

von Hippel, E. (2005). *Democratizing innovation*, MIT Press.

Zjawinski, S. (2007). Framing open source architecture, `http://www.wired.com/culture/design/news/2007/03/72902?currentPage=all`, ultima visita novembre 2013.