Introduction: SimpleHelp, a company that provides remote support software, is expanding its product range to include a new product. The new product, “SimpleGrid”, will enable clients to set up private computer grids that perform computation-intensive tasks by distributing and running tasks on underutilised machines at any moment. As the product will provide supercomputer-like capabilities to users who may not have used such systems before, the quality and usability of the interface are crucial. Our task is to design an interface that will enable the sophisticated configuration of a potentially complex distributed software-hardware system while making the process and its monitoring simple enough to make the configuration of new tasks and grids worthwhile.

Design approach: Design for the user paradigm

Challenges: We designed what we thought was the best for the system. However, we failed to involve users in our design. This design paradigm will cause company configuration errors, the need for extensive user training, and usability issues.

Lesson learned: Always involve end users in your design.

Authors: Adamu Adamu Habu, Iain Carson and Lan Liu.

SimpleGrid

UI Design

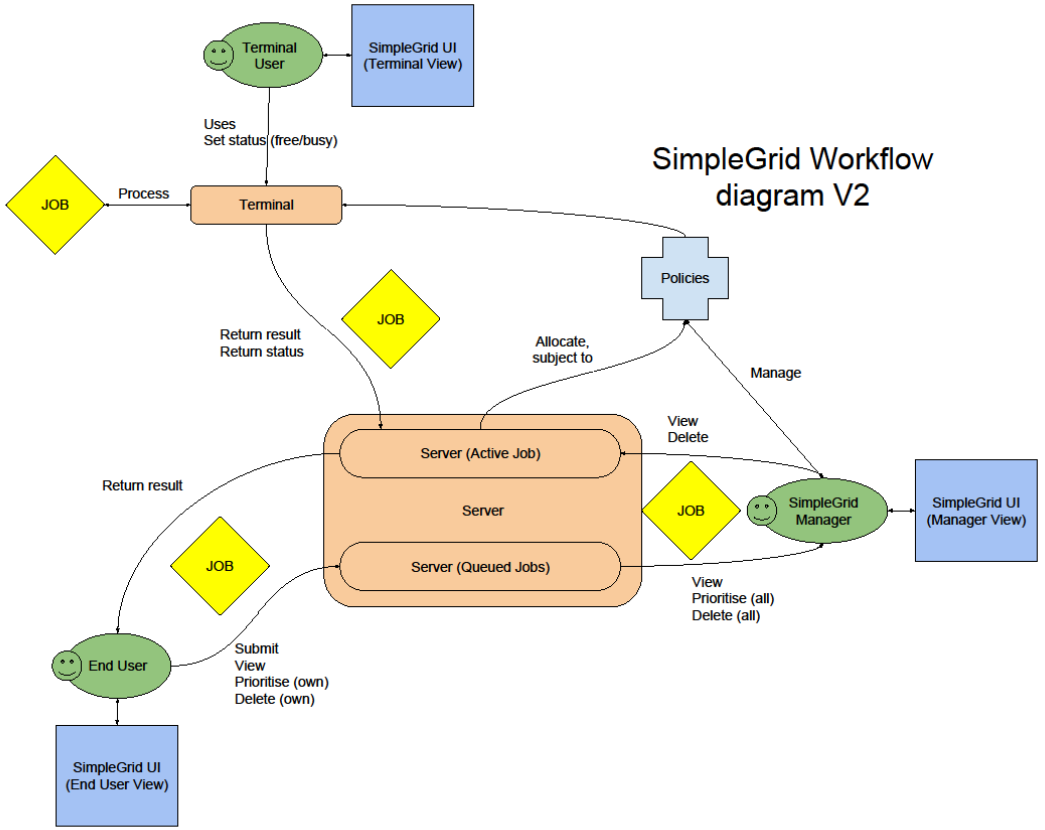
Contents

- Scenario
- Scope
- Design elements
- Interfaces

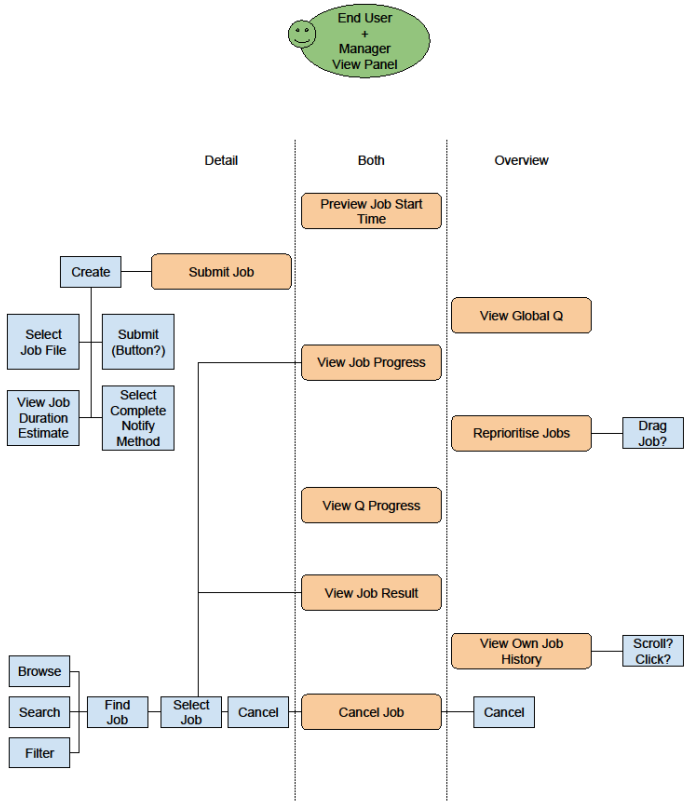
Scenario


NETWORK: 50%
QUEUE :
JOB 1
JOB 2
JOB 3
SUBMIT JOB
CLICK
JOB COMPLETE!
100%
RESULT...

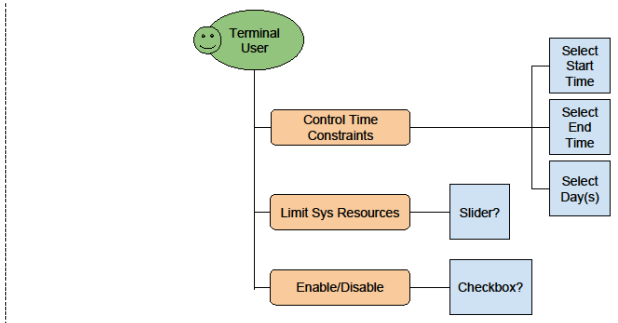
Scope: Workflow Diagram

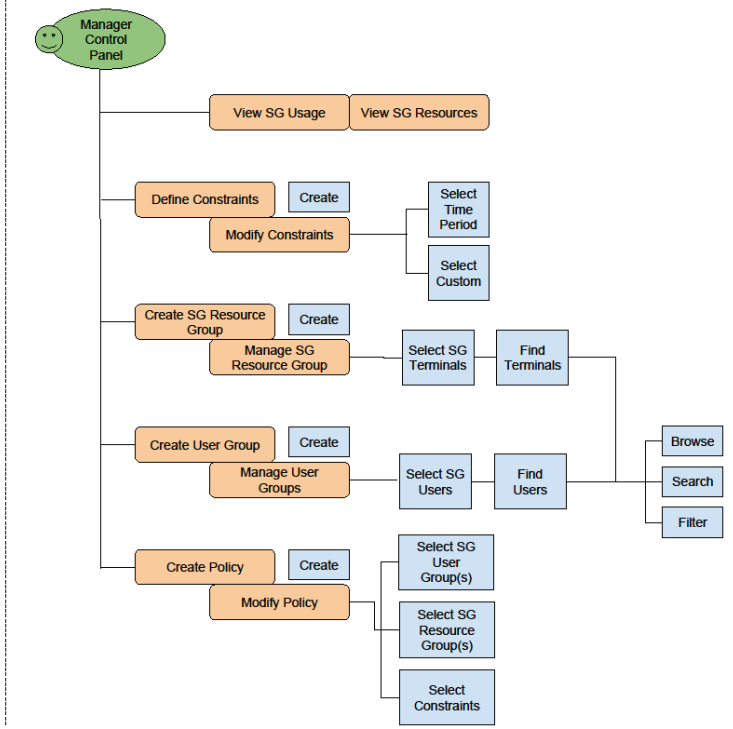


Scope: Requirements Diagram

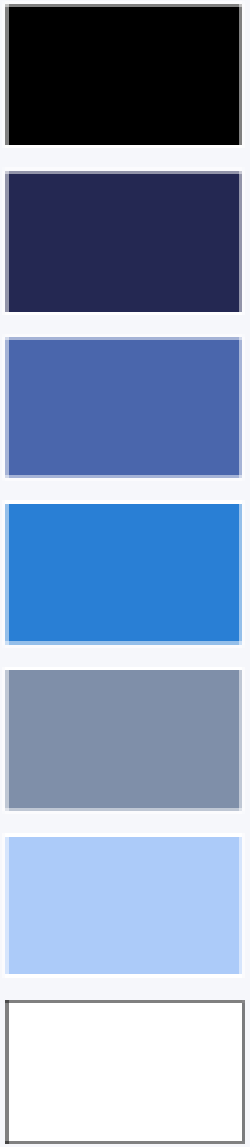


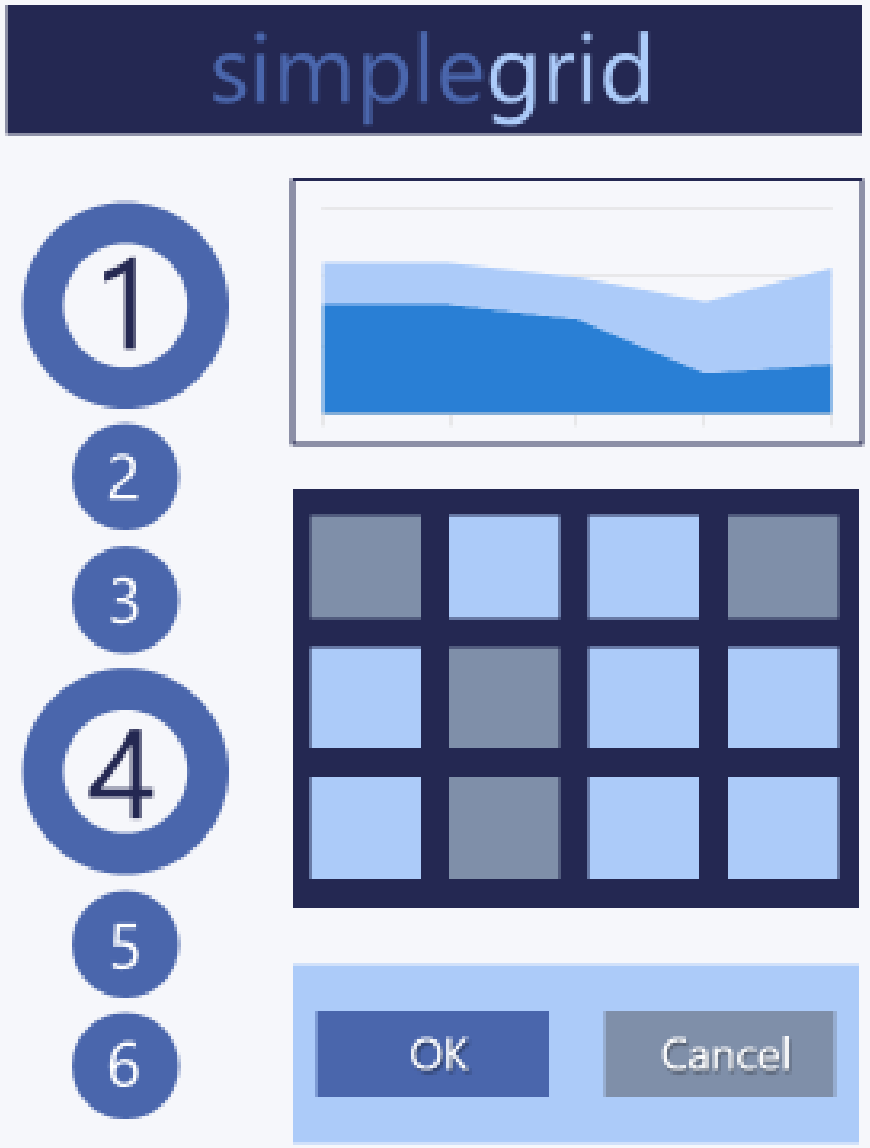
SimpleGrid Reqs
Overview V1

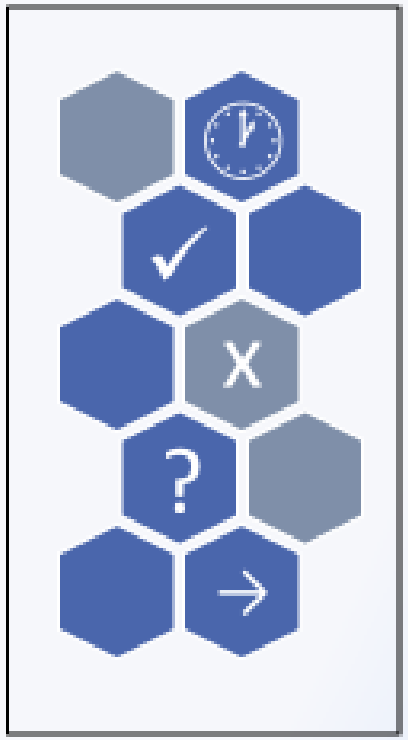




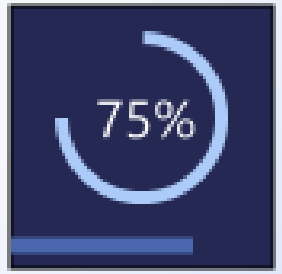
Moodboard

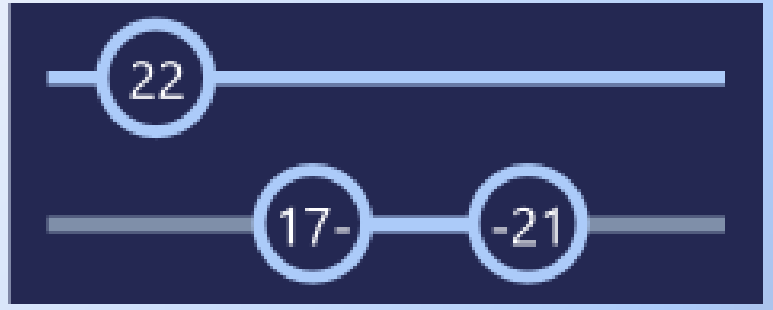






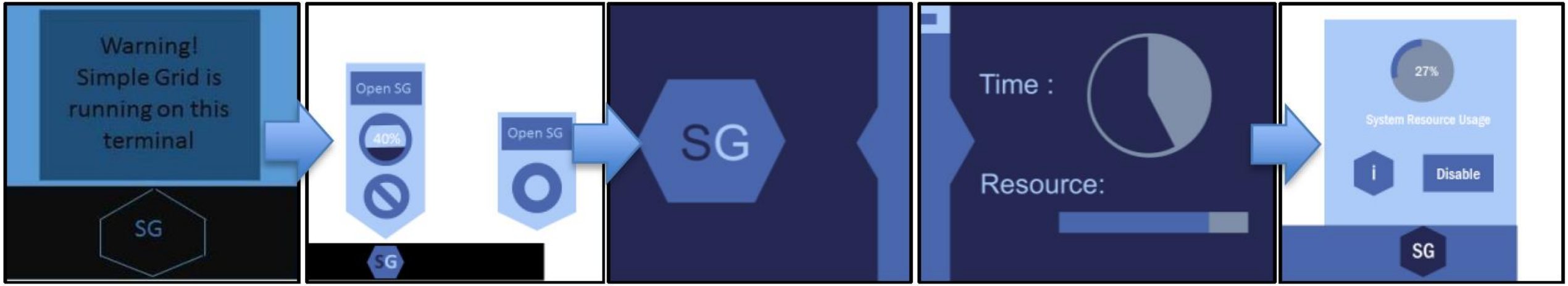
Mo	Tu	We	Th	Fr	Sa	Su
	1	2	3	4	5	6
7	8	9	10	11	12	13
14	15	16	17	18	19	20
21	22	23	24	25	26	27
28	29	30				



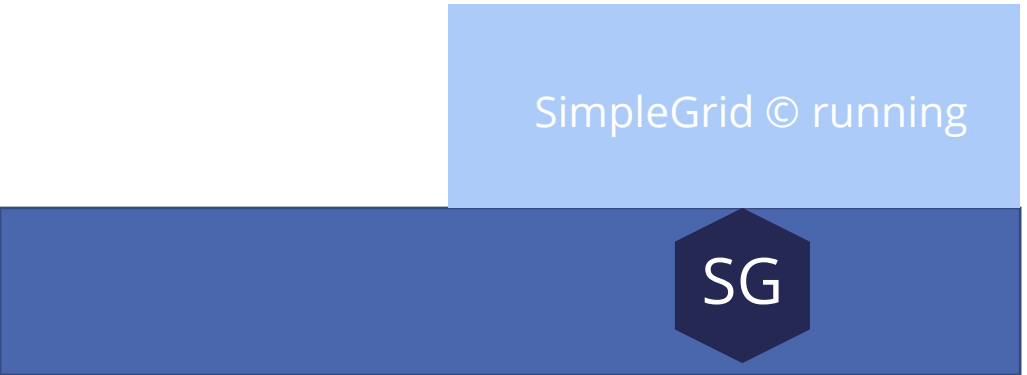


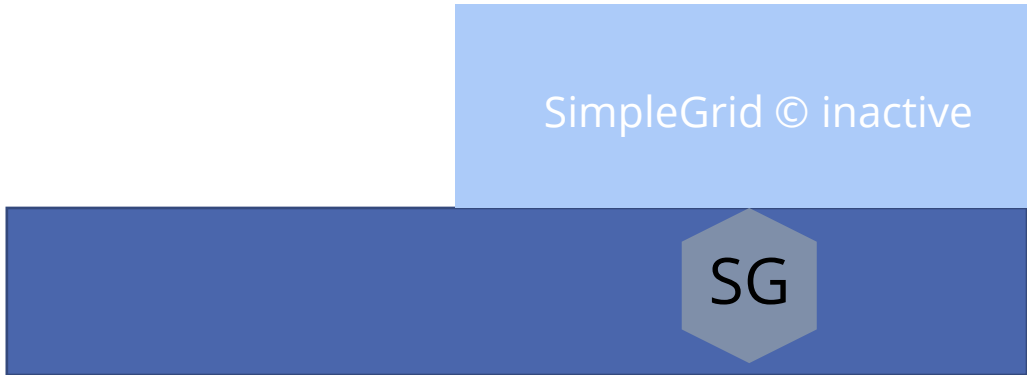
Sketching

- Design
- Process



Terminal User

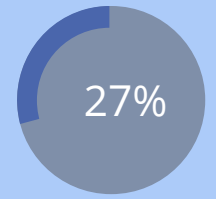

SimpleGrid © running
SG


SimpleGrid © inactive
SG

SimpleGrid ©
2017

SimpleGrid runs on this terminal to allow computation of important work-related jobs.

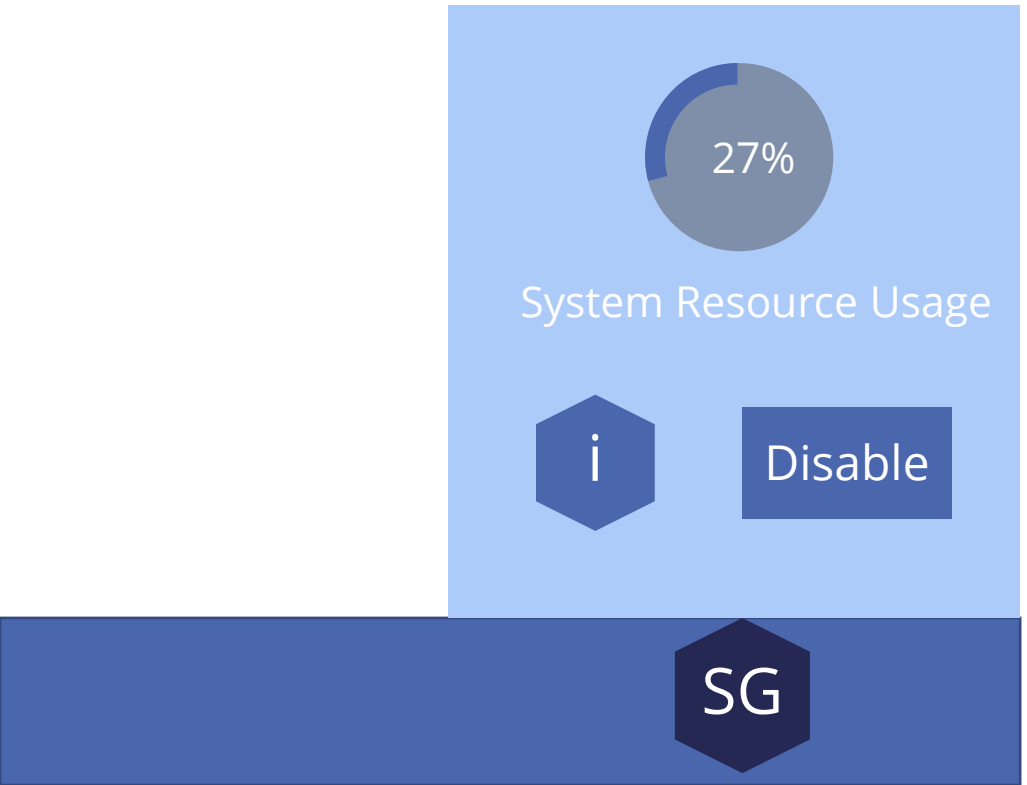
Contact your network administrator for more information.

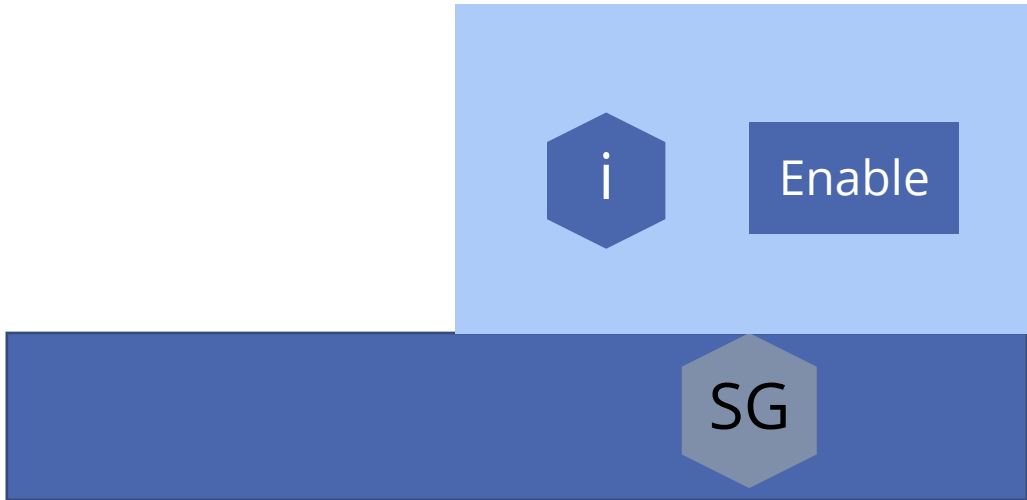


System Resource Usage

i

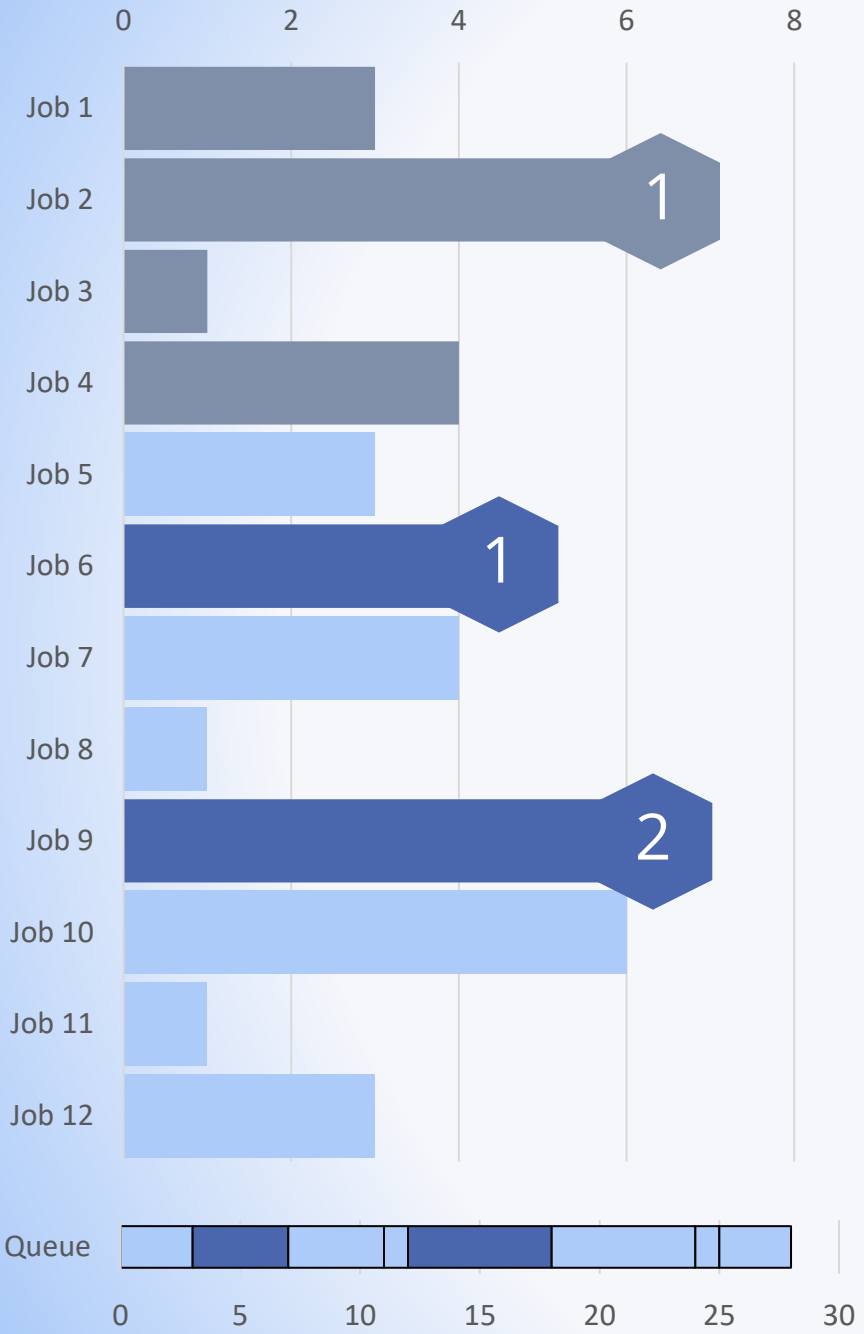
SG


27%
System Resource Usage
i
Disable
SG


i
Enable
SG

End User

Simplegrid – gridUser



Submit New Job

/path/to/job.file Browse +

Active Jobs

		Submitted	Est. Complete	Delete
1	75%	16/04/2017 14:22	18/04/2017 18:30	X
2	0%	16/04/2017 16:40	20/04/2017 11:30	X

Completed Jobs

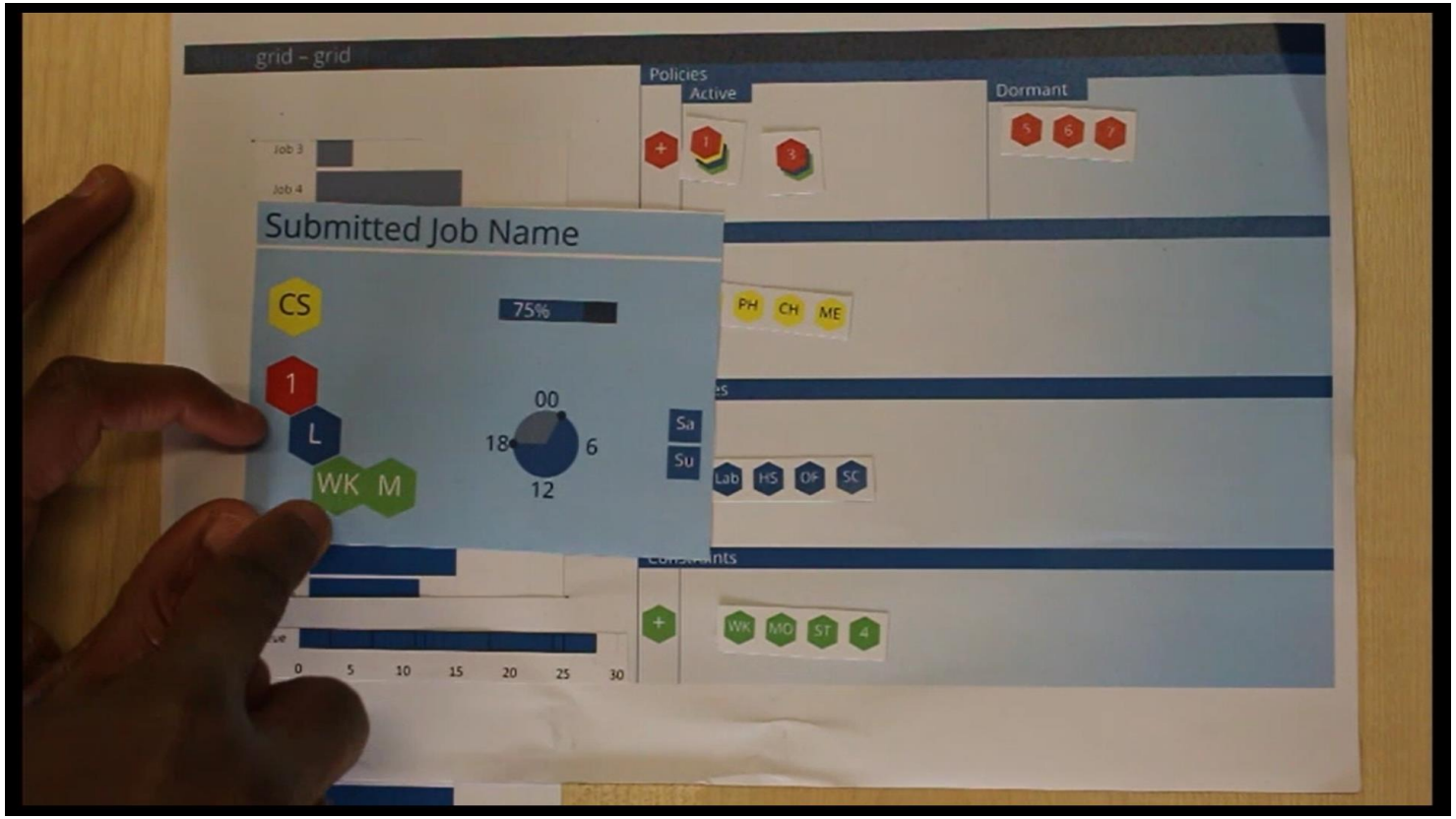
		Submitted	Completed	Details
1	75%	12/04/2017 09:10	13/04/2017 12:00	→

Network Manager

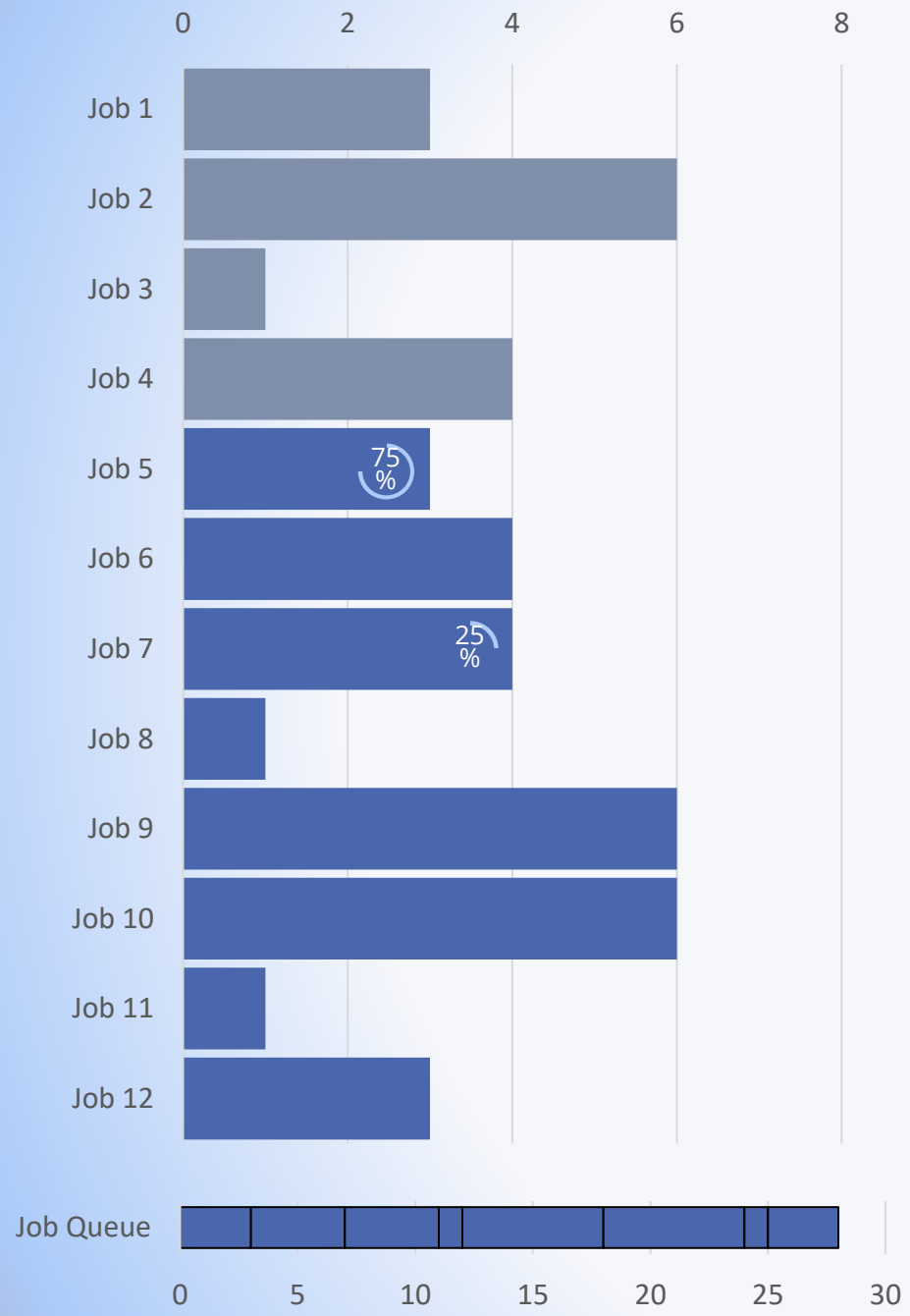
Interactive Prototype

- Job queue
- Policy creation tool
  - User Groups
  - Resource Groups
  - Constraints

- Guide video provided



Simplegrid – gridManager



Policies

Active

1 3 7 8

Dormant

2 4 5 6

Groups

CS PH CH ME

Resources

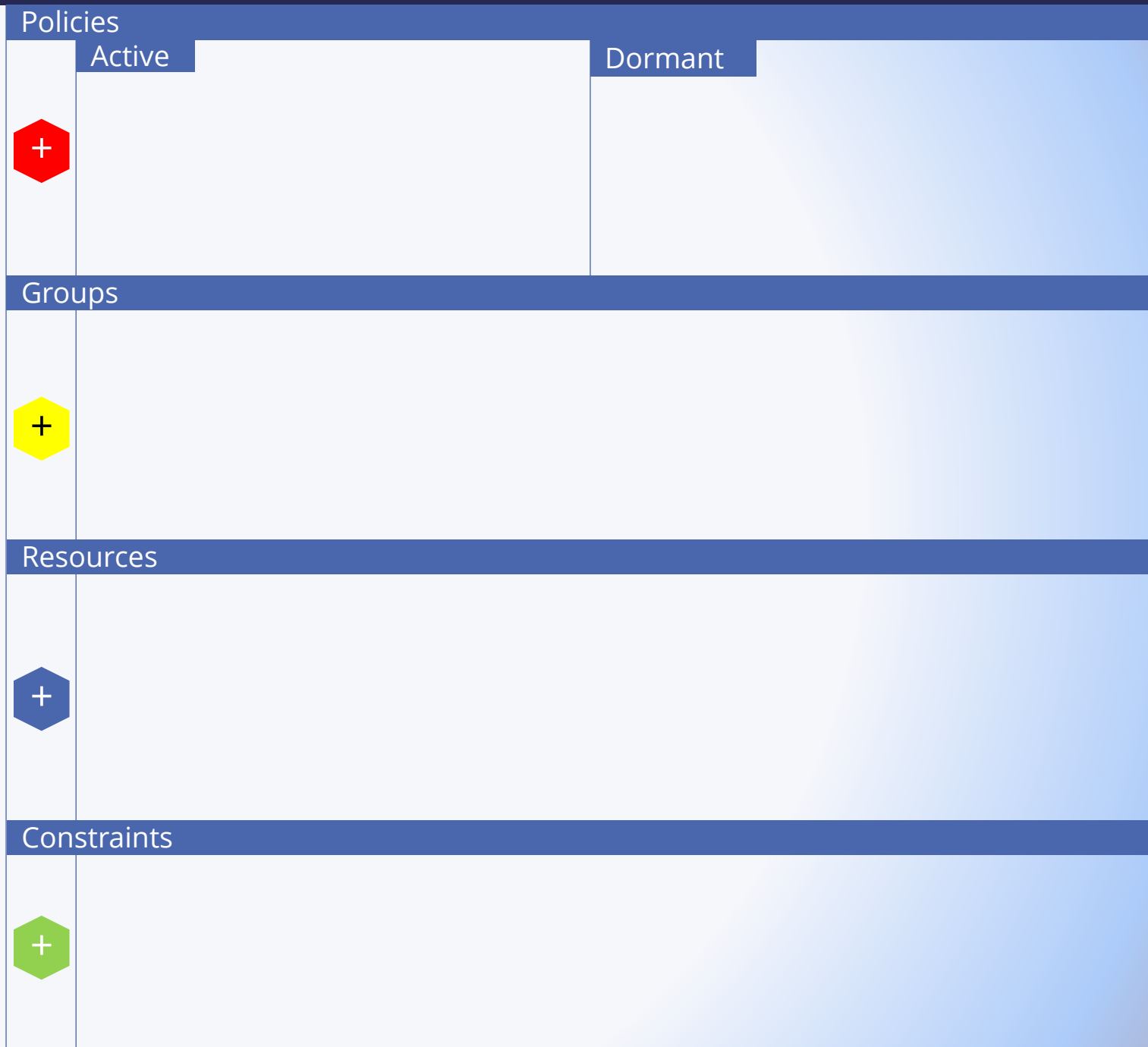
Lab HS OF SC

Constraints

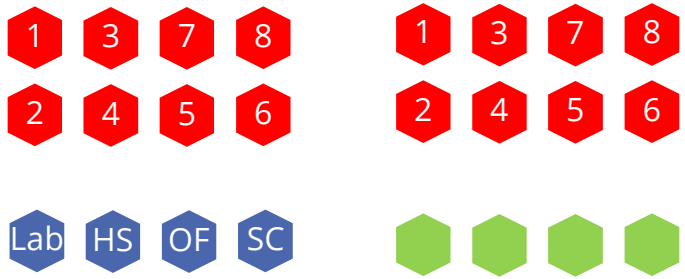
WK MO ST 4

Simplegrid – gridManager

Simplegrid – gridManager

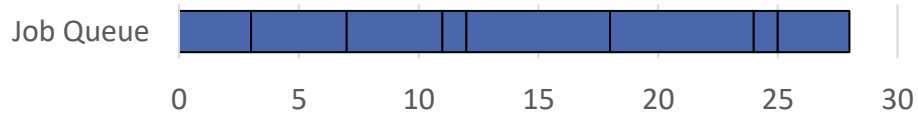


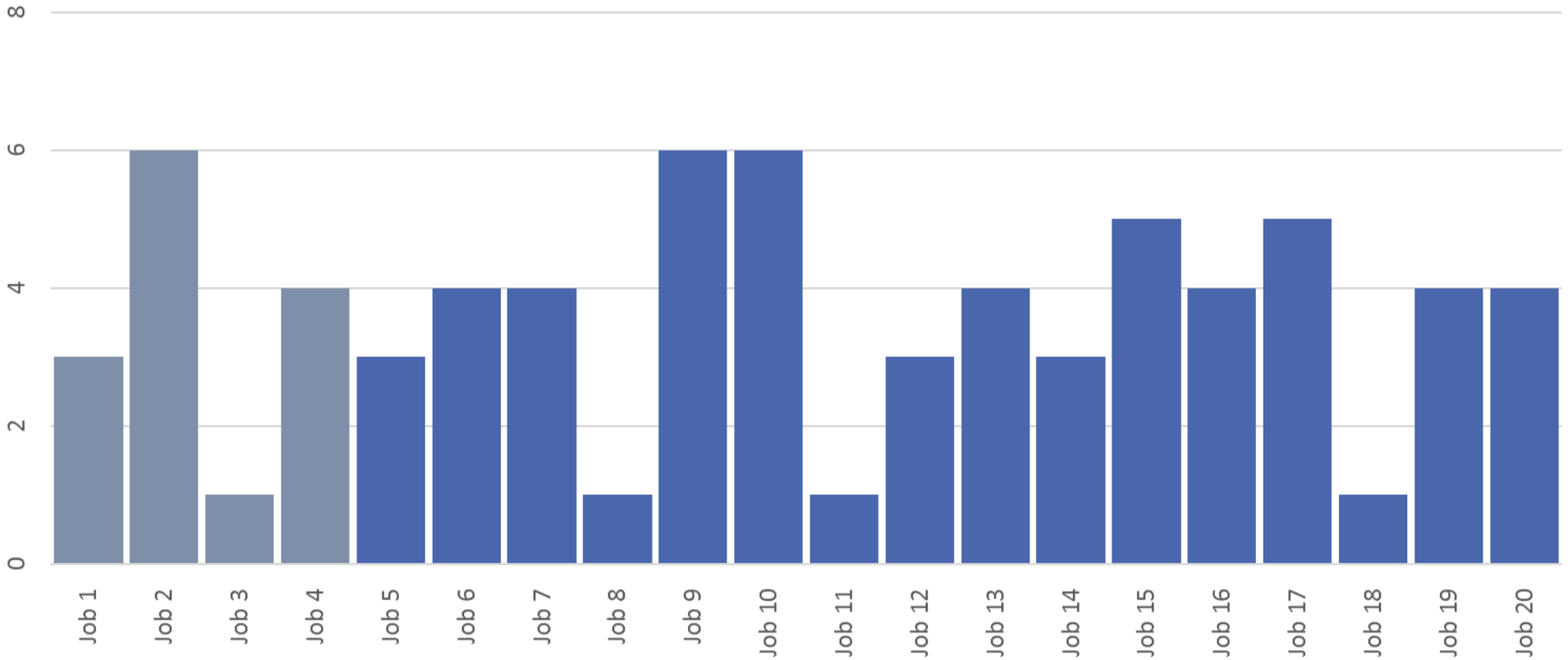

CS
PH
CH
ME
CS
PH
CH
ME
Lab
HS
OF
SC
Lab
HS
OF
SC
WK
MO
ST
4

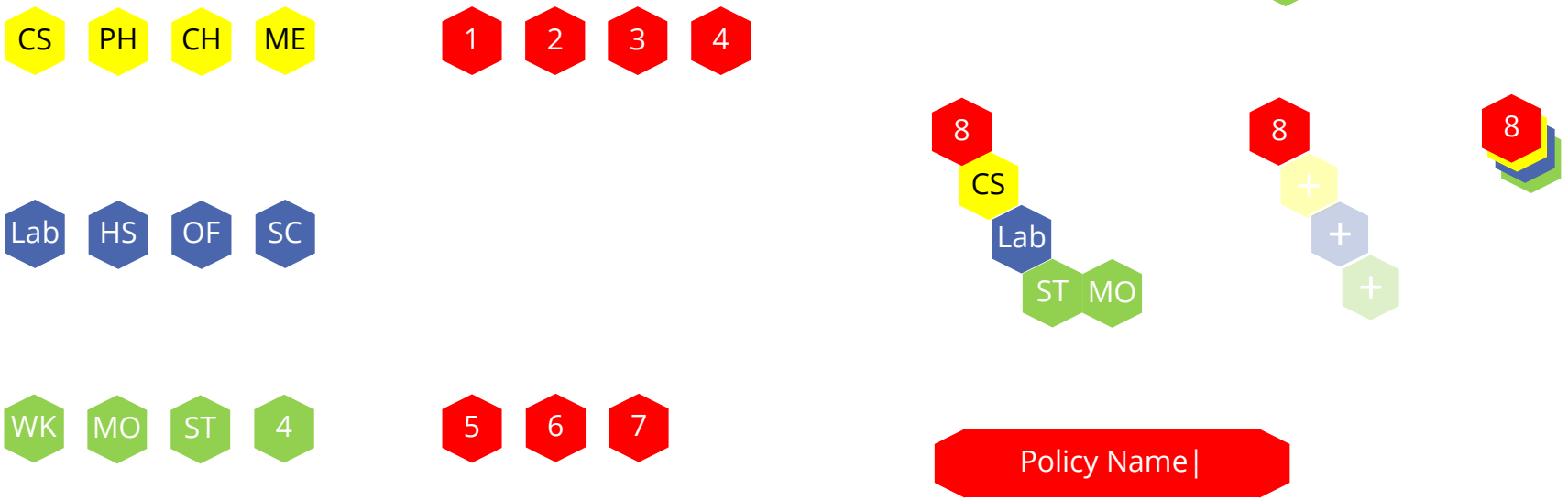

1
3
7
8
2
4
5
6
Lab
HS
OF
SC

1
3
7
8
2
4
5
6

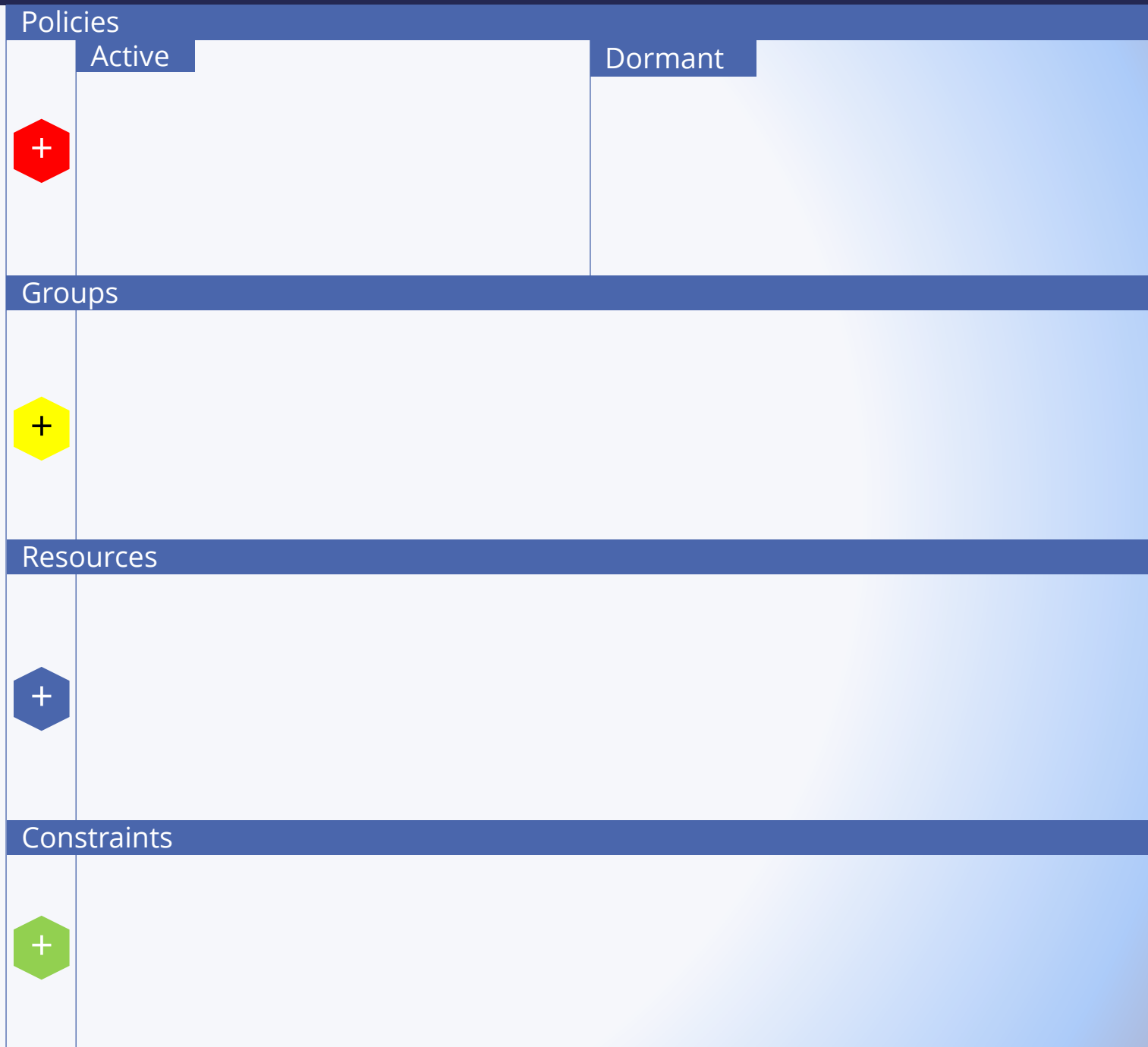

1
3

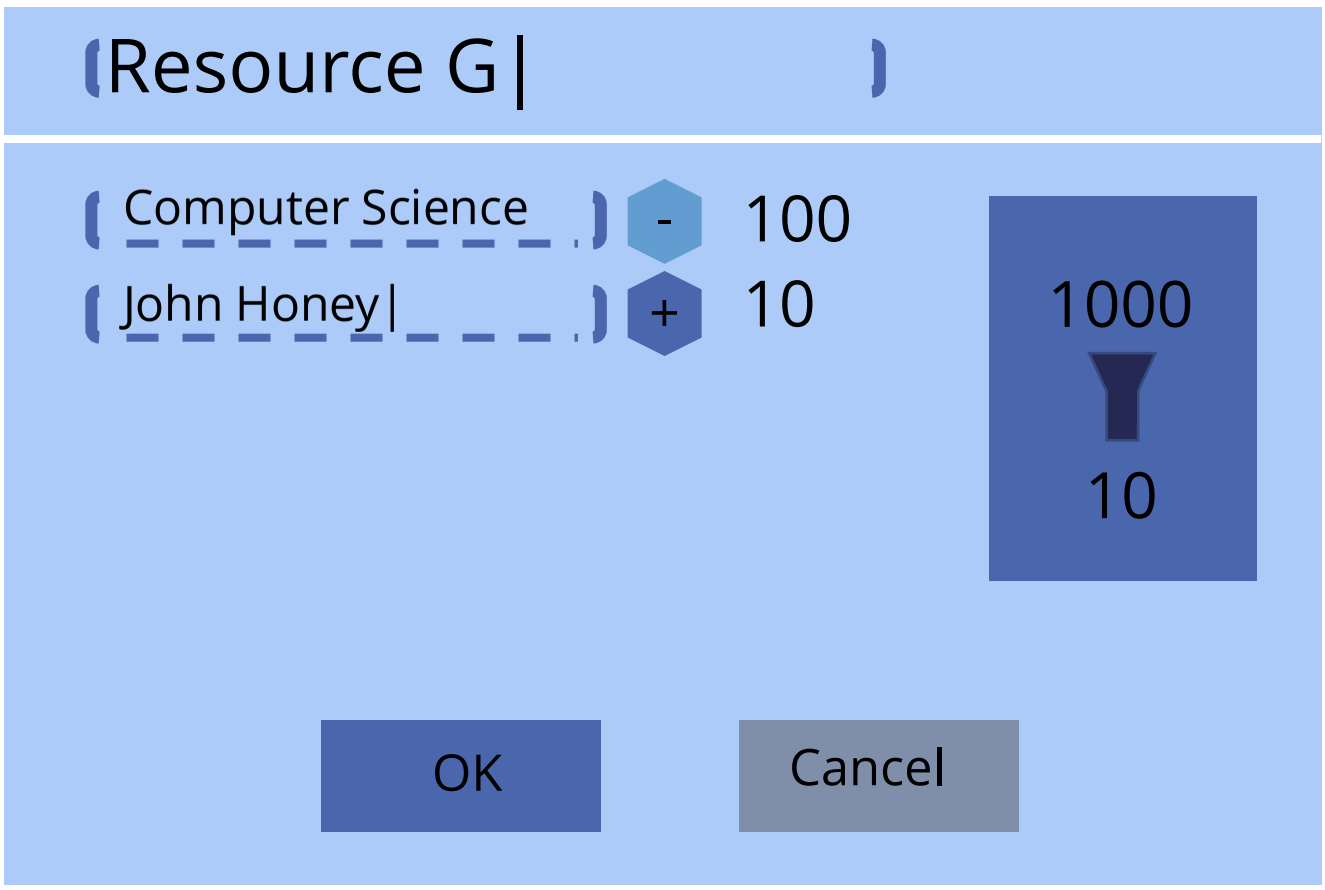

Job Queue
0
5
10
15
20
25
30

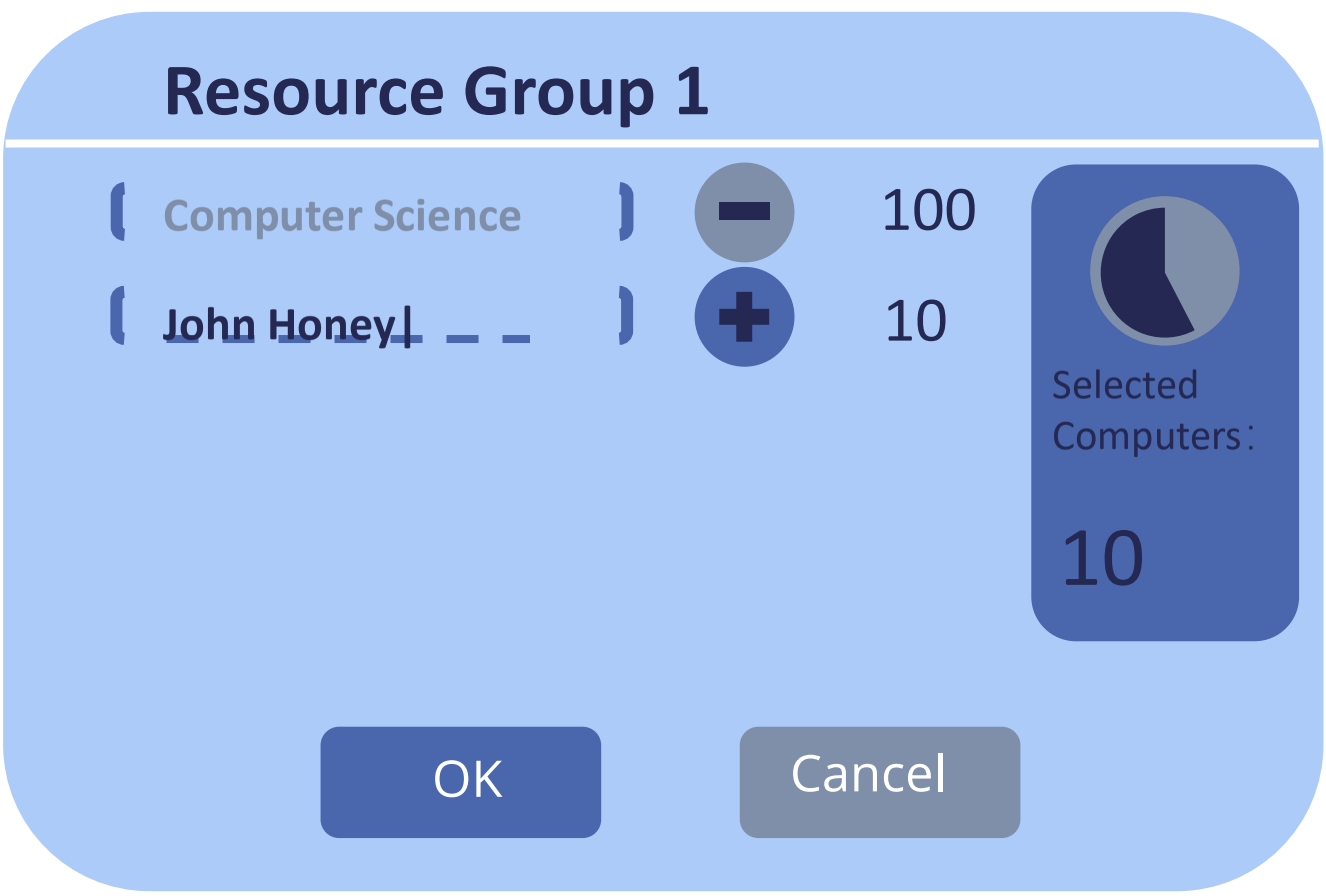

8
6
4
2
0
Job 1
Job 2
Job 3
Job 4
Job 5
Job 6
Job 7
Job 8
Job 9
Job 10
Job 11
Job 12
Job 13
Job 14
Job 15
Job 16
Job 17
Job 18
Job 19
Job 20


CS
PH
CH
ME
1
2
3
4
Lab
HS
OF
SC
WK
MO
ST
4
5
6
7
1
3
8
CS
Lab
ST
MO
8
+
+
+
8
Policy Name|

Simplegrid – gridManager

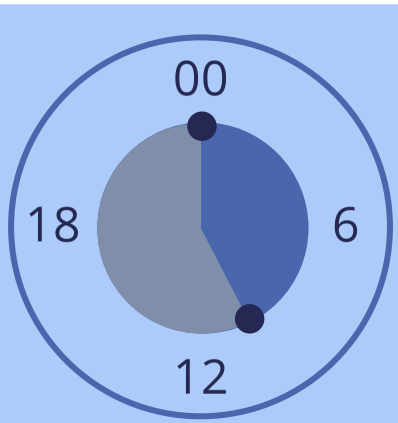



Resource G
Computer Science
-
100
John Honey
+
10
1000
10
OK
Cancel


Resource Group 1
Computer Science
100
John Honey
10
Selected Computers:
10
OK
Cancel

Constraint N|

Time :



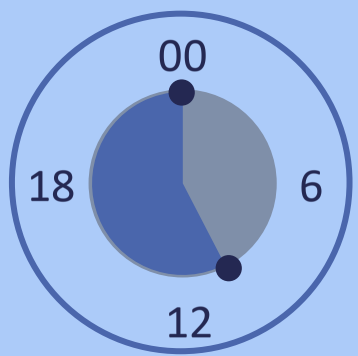
From: 00am
To: 10am

Day : Mo Tu We Th Fr Sa Su

OK Cancel

Constraint Nam|

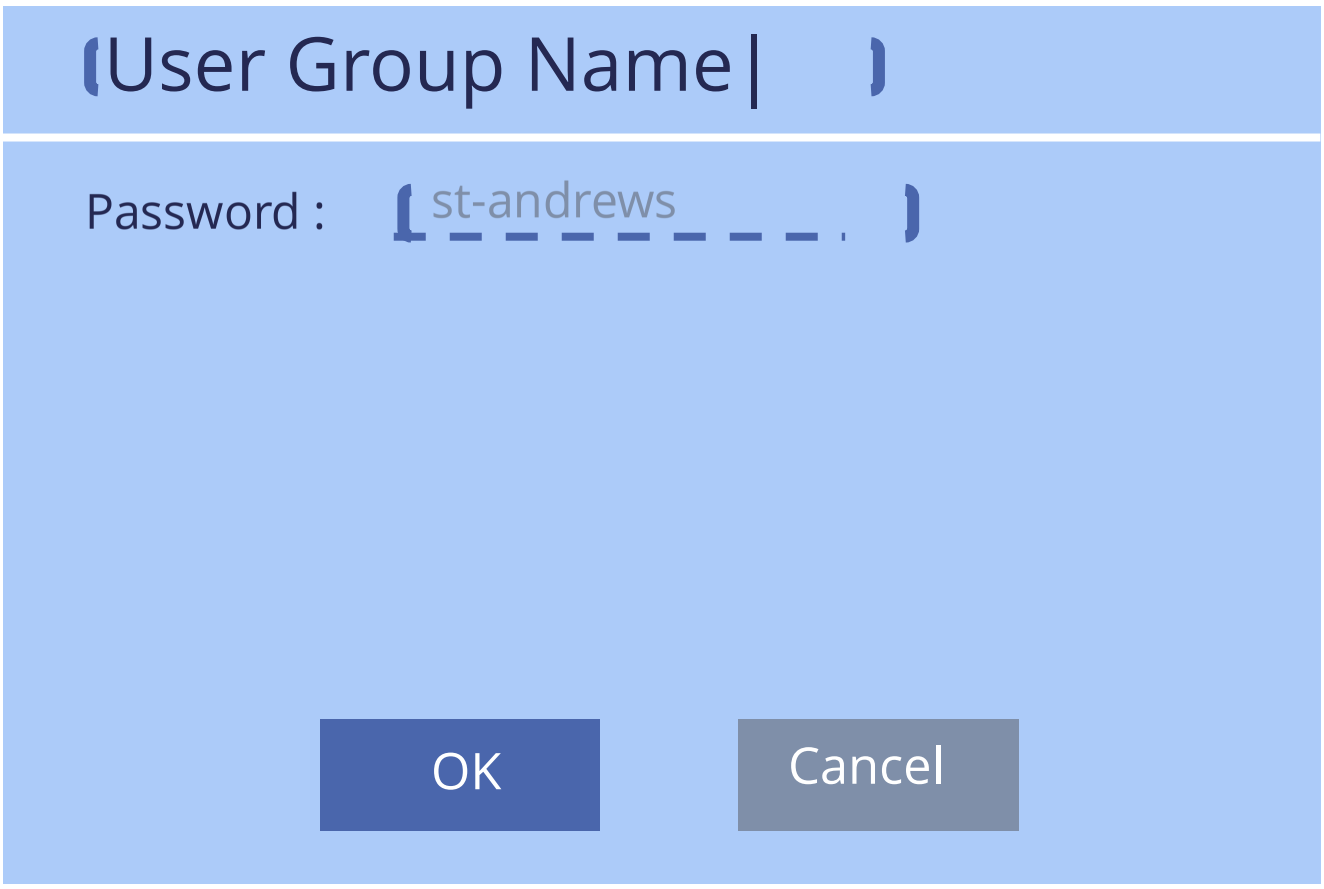
Time :

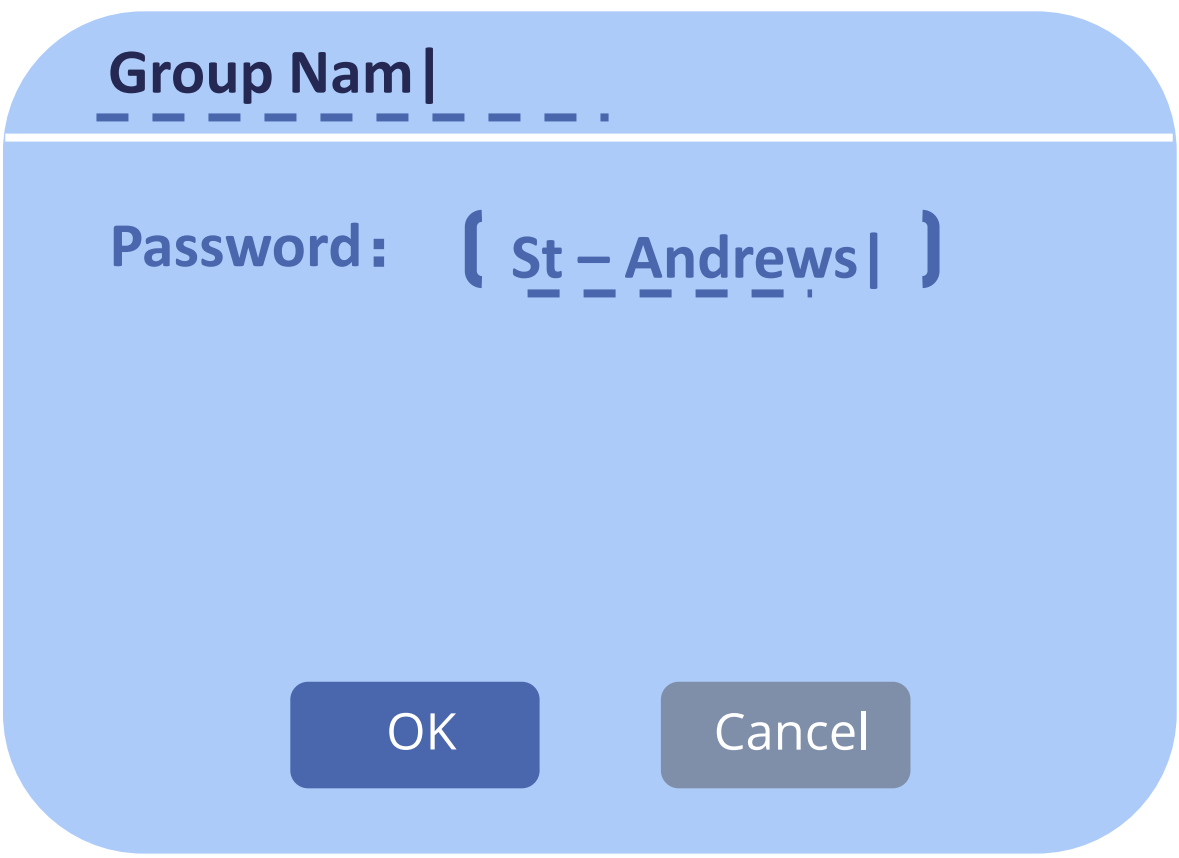


From: 00am
To: 10am

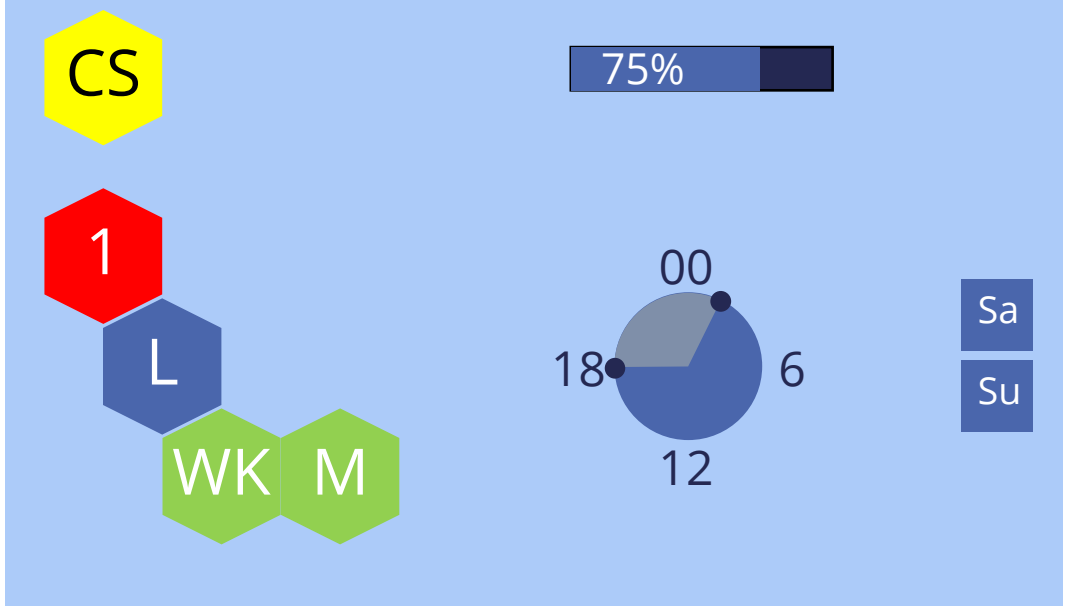
Day : Mo Tu We Th Fr Sa Su

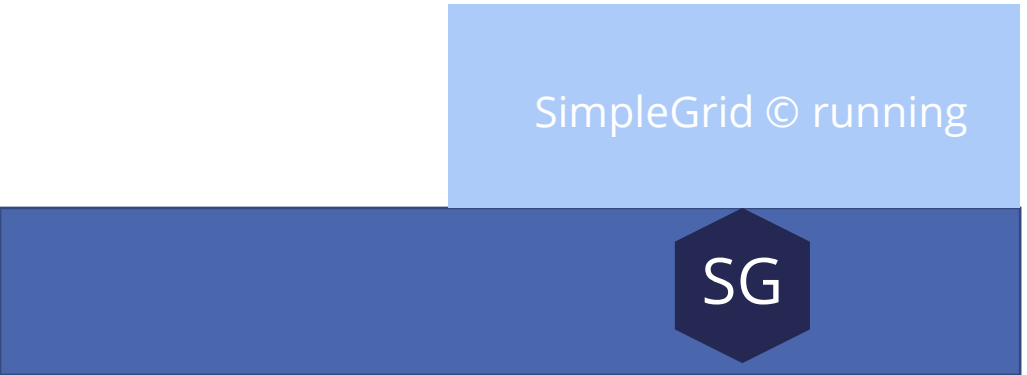
OK Cancel

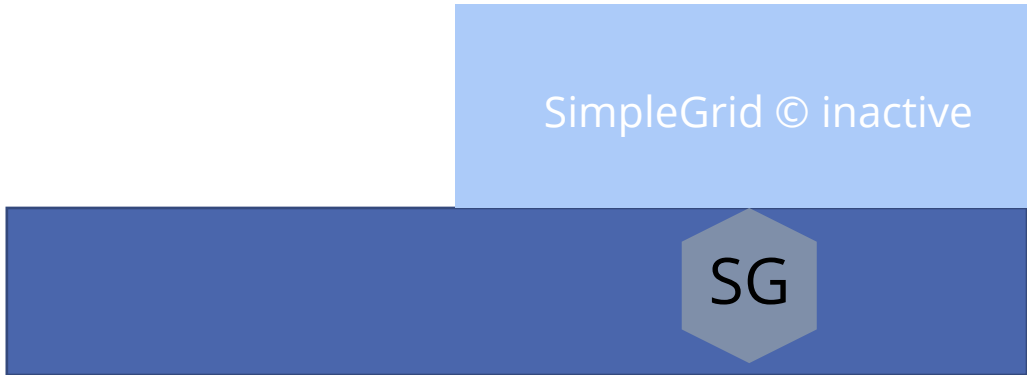

User Group Name
Password :
st-andrews
OK
Cancel

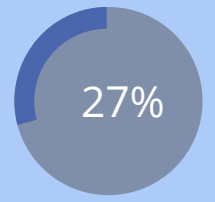

Group Nam
Password :
St – Andrews
OK
Cancel

Submitted Job Name




SimpleGrid © running
SG


SimpleGrid © inactive
SG



System Resource Usage

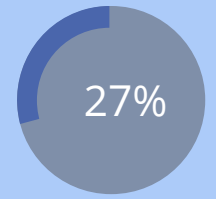
i

SG

SimpleGrid ©
2017

SimpleGrid runs on this terminal to allow computation of important work-related jobs.

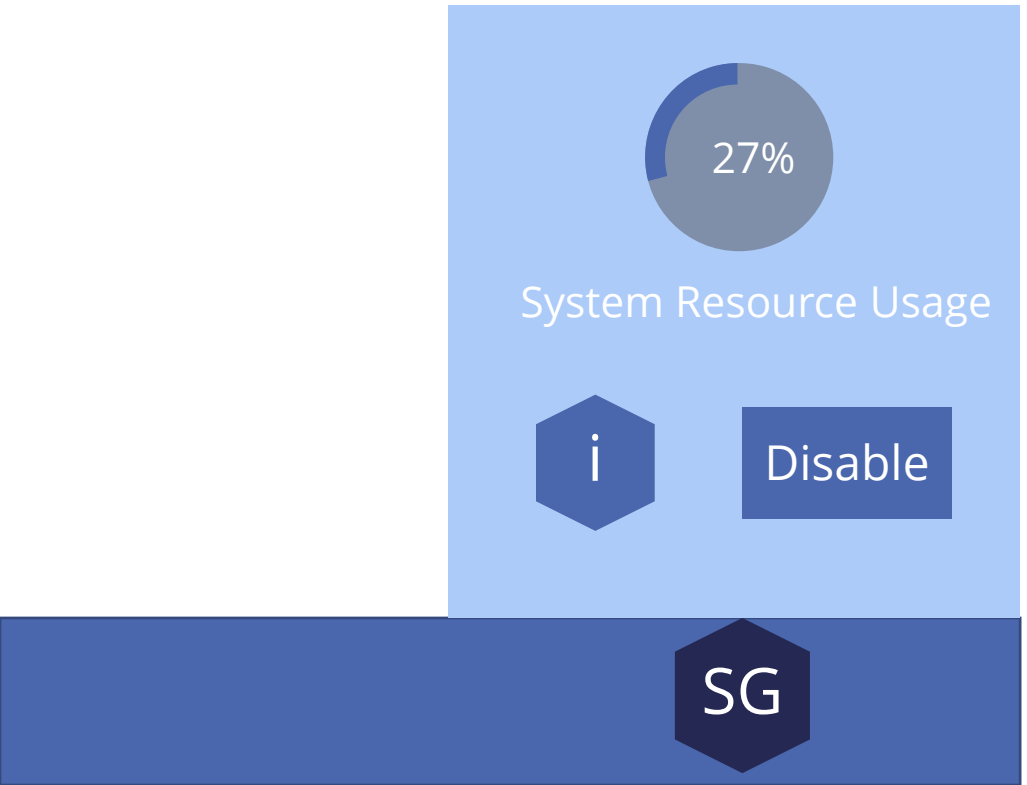
Contact your network administrator for more information.

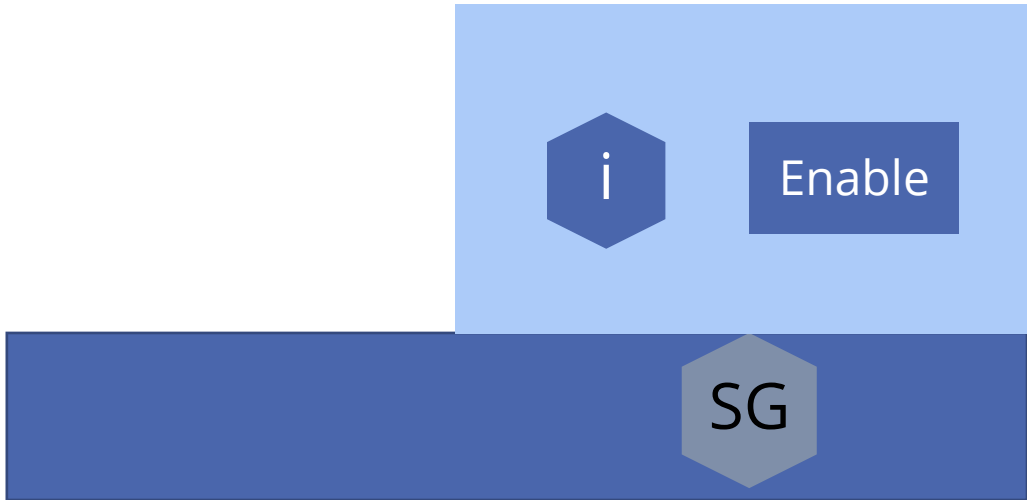


System Resource Usage

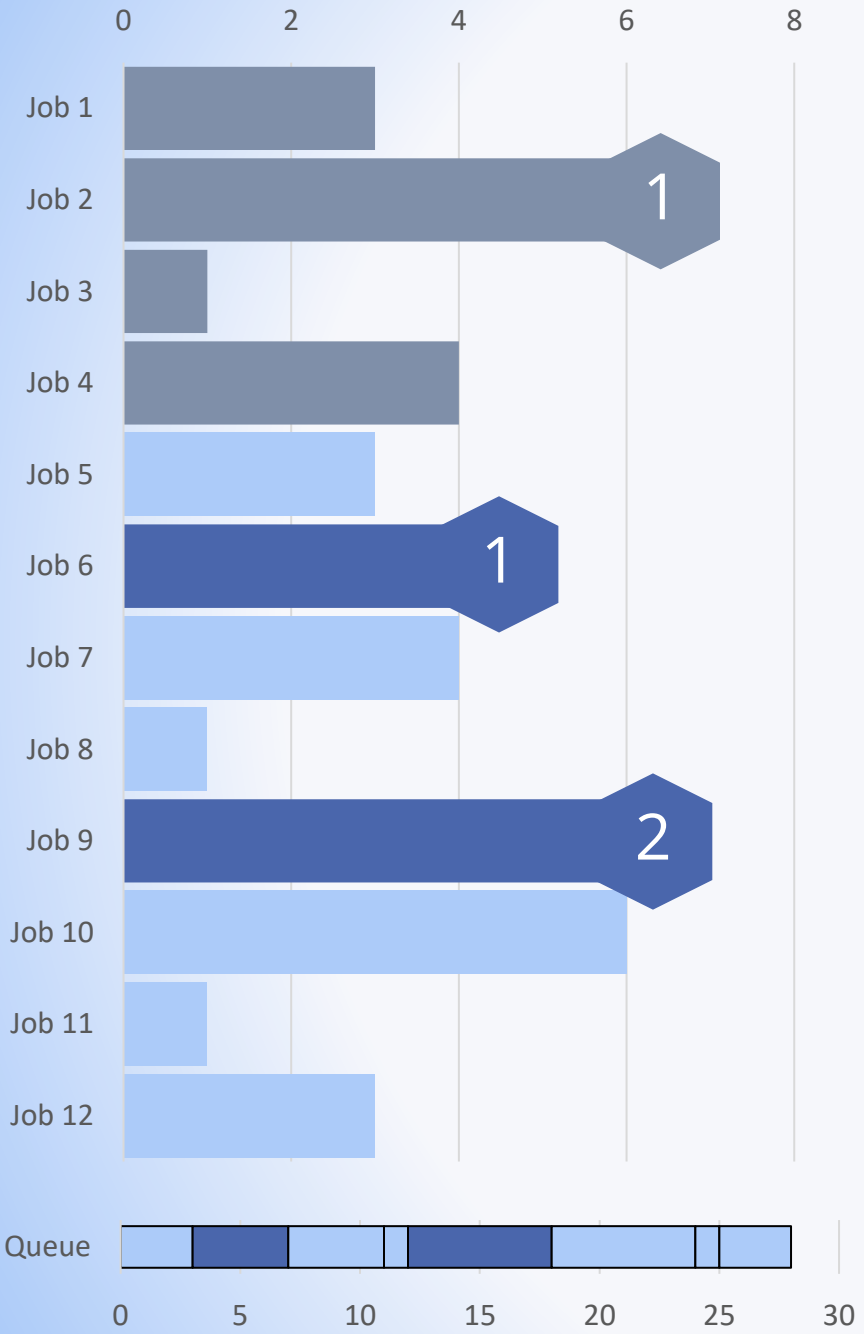
i

SG


27%
System Resource Usage
i
Disable
SG


i
Enable
SG

Simplegrid – gridUser



Submit New Job

/path/to/job.file Browse +

Active Jobs

		Submitted	Est. Complete	Delete
1	75%	16/04/2017 14:22	18/04/2017 18:30	X
2	0%	16/04/2017 16:40	20/04/2017 11:30	X

Completed Jobs

		Submitted	Completed	Details
1	75%	12/04/2017 09:10	13/04/2017 12:00	→

CS5042 - P1
SimpleGrid

Iain Carson
University of St Andrews
ic48@st-andrews.ac.uk

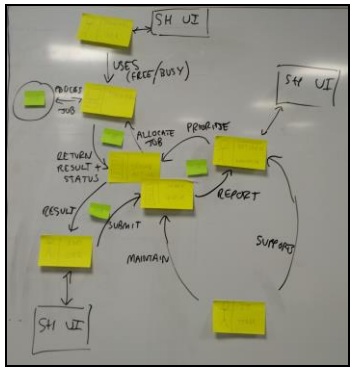
Adamu Adamu Habu
University of St Andrews
aah5@st-andrews.ac.uk

Lan Liu
University of St Andrews
ll78@st-andrews.ac.uk

Introduction

SimpleHelp, a company that provides remote support software, are expanding their product range to include a new product. The new product, named "SimpleGrid", will enable clients to set up private computer grids that perform computation-intensive tasks by distributing and running tasks on machines that are underutilized at any particular moment. As the product will be providing supercomputer-like capabilities to users who may not have used such systems before, the quality and usability of the interface is crucial. Our task is to design an interface which will enable sophisticated configuration of a potentially complex distributed software-hardware system while making the process and its monitoring simple enough to make the configuration of new tasks and grids worthwhile.

Work Flow Diagram



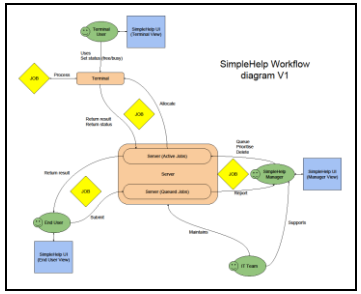


Figure 1: The initial Work Flow Diagram and its associated digitised version

Contextual Enquiry

Initial contact with the client

Our first step was to establish contact with the client. Through email we arranged a meeting with our local contact Ian Miguel, followed by a Skype meeting with SimpleHelp themselves. A brief group meeting was held prior to contact, to plan the agendas for both meetings. It was agreed at this early stage that all communication with the customer would be carefully planned so as not to waste their time.

We introduced ourselves to Ian and verified the various roles and best methods of communication. Following this, our arranged Skype meeting with SimpleHelp served three purposes:

1. To introduce ourselves and meet SimpleHelp's representatives
2. Establish best methods of communication with SimpleHelp
3. Gain a basic understanding of the product and SimpleHelp's expectations of the project.

From the information gathered in this first contact, we worked to generate a basic work flow diagram.

Flow models, and task-structure models

Our initial conversation highlighted the existence of three actors in the SimpleGrid system, and we agreed on system-specific terminology that would allow the team and client to succinctly communicate ideas regarding the users and interfaces in the project. First, "Terminal Users" (TU); those whose terminals are provided as resources for jobs to be performed on. Next, those who create, upload and receive the results of jobs were titled "End Users" (EU). Finally, a "SimpleGrid network manager" (NM); an individual or group who manages SimpleGrid and rectifies basic system errors. The SimpleGrid network manager interacts with the SimpleGrid server (provided by the client) to add and adjust protocols which manage task execution order and details.

After labelling the main actors, we identified other entities in the system, including the terminals and server. Labelling and sticking post-it notes to a whiteboard, we drew curves and arrows to link different entities and actors, and illustrate the information transfer between different entities, thus generating Figure 1; a work flow model to help us better visualise the system. Sticky notes allow us to modify the number and location of entities as we learn and discuss within the group and with the client. SimpleGrid runs entirely on a local network without external actors, eliminating the need for the representation of external roles in the diagram. Two interesting insights were gained at this point: first that we would likely have to consider creation of three user interfaces, and second that as it is a "closed" system that is still in development, we have a lot of freedom in terms of design.

A digitised version of the diagram was emailed to SimpleHelp for confirmation that we had understood their descriptions, and their feedback was used to further update the work flow diagram both on the board and digitally.

Design of the Contextual Enquiry

Analysis of the amended Work Flow diagram revealed the overall structure and flow of information in the SimpleGrid network, but did not tell us about individual

tasks. For this we designed a contextual enquiry to elicit user tasks, with the end goal of generating an interface requirements document.

(Hartson & Pyla, 2012) suggest that the optimal method of task elicitation is through observation or interview of current system users. However, SimpleGrid does not yet exist, and even at a basic level is still under construction. (Hartson & Pyla, 2012)'s next-best suggestion is to observe a system that performs similar tasks, which in SimpleGrid's case is a supercomputer.

An observation of clients using similar systems (to gather work tasks and gain insights into process models) was therefore considered, though was not implemented for two major reasons: First, as the software we are designing for is quite novel in its functionality (as well as being incomplete) we have wide scope for innovation and creativity in the way users will interact with it. Observing current processes of similar systems may guide design through awareness of current standards, but at the cost of creativity in our own process design. Secondly, all the interfaces of current systems are command-line-based, and utilised by mathematicians and computer scientists who would not necessarily want to use GUI applications. This paradigm is something we seek to move away from in implementing the SimpleGrid UI. With these in consideration, it was decided that performing an observation of a similar system would be of more harm than good.

Instead, a semi-structured interview with the client was chosen as this could be carefully designed to elicit practical requirements whilst reserving judgement regarding design. Interviews also have the advantage of being relatively easy to perform remotely (via Skype for example) and this had proven true in the previous customer conversation.

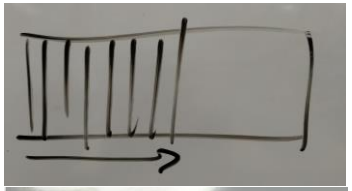
The client was sent the relevant, completed ethics documents, and upon signing and returning them we confirmed the time for the interview. The customer's permission to record the interview was asked and approved.

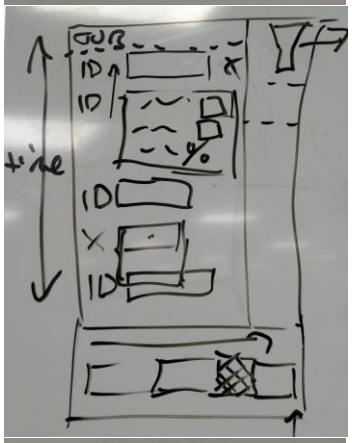
The interview conducted was structured to achieve three goals. Primarily, the first part of the interview explored the hardware of the final user interface, so we knew what presentation format to design for. The next part of the interview focussed on clarification of holes in the Work Flow diagram, mainly revolving around the role of "constraints" in the system. Knowing that the next step in the process would be creation of the WAAD for consideration of task relations and organisation, the final part of the interview worked systematically through the various tasks foreseen by the team for each user interface, starting with the simplest interface (TU), through to the most complex (NM). With the full UX design process (Hartson & Pyla, 2012) in mind, the questions asked in interview were designed to elicit responses that could be easily translated into Work Activity notes (WANs - which play a vital role in contextual analysis) if not directly into requirements.

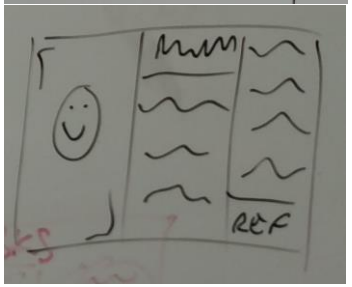
An example of a question asked in the interview would be "On the server, jobs will be prioritised somehow - what, in your opinion influences the prioritisation process?". The anticipated answers to this question highlight the factors we need to give various users control of in each interface.

Insight Development

During the entire process, group members were encouraged to illustrate ideas and present insights on the whiteboard for discussion:







Contextual Analysis

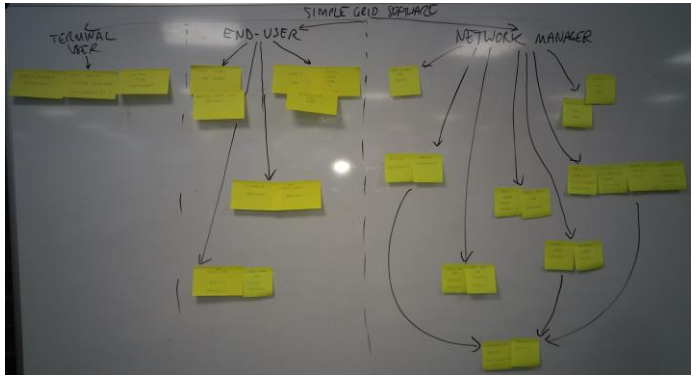
Transcription and Work Activity Notes (WANs)

The audio recording and the notes taken by each team member during interview were transcribed. The raw data was analysed and the key points, issues and ideas were captured. For example, the answers to the above question elicited the key points of "Constraints, submission time, network resource limits, user logged on/system resources, size of job/data usage, which end user it came from", which allowed development of ideas, such as "Constraints will be used to manage when a job is processed".

The notes were worked through systematically, and combination of ideas and key points allowed WANs to be created. Some example WANs include:

"SimpleGrid Manager should be able to define constraints and modify constraints". This WAN can also be directly transcribed into a system requirement.

The initial flow model was refined and corrected where necessary.

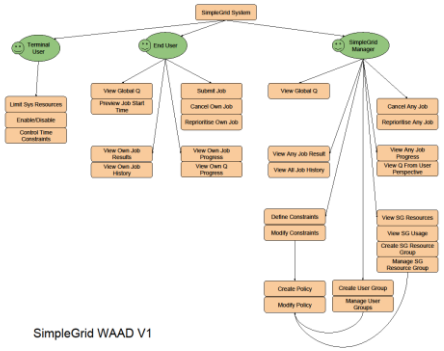


Work Activity Affinity Diagram (WAAD)

According to (Hartson & Pyla, 2012) the next stage in Contextual Analysis after generating the WANs in the contextual analysis is the creation of a Work Activity Affinity Diagram. The WAAD is a tool employed to allow grouping and organisation of WANs in a visual display format, helping to consolidate and generalise the data gathered from the Contextual Enquiry. In order to maintain a permanent location for the WAAD, a single board in the John Honey Lab was cleared. The WANs were written onto sticky notes.

A session leader was appointed, and each member of the team picked some WANs at random. Members read aloud the work activity note in their hand, one at a time, and after deliberation, the stickers were placed in the agreed location on the board. Similar WANs were posted on the board together.

Initially, the WANs were grouped and organised in hierarchical order according to interface (TU, EU, NM) as this was the most intuitive and obvious arrangement, resulting in the WAAD shown in Figure 2.



SimpleGrid WAAD V1

Figure 2: The original WAAD and its digital counterpart.

WANs were written onto post-it notes and arranged by the group collaboratively

Revised WorkFlow Diagram

As recommended by (Hartson & Pyla, 2012), the Work Flow diagram was continually revised in response to new information and insights.

The below shows the amendments made following the first interview, namely inclusion of "constraints" into our diagram, and simplification of the SimpleGrid Network Manager to implicitly suggest presence within an IT team.

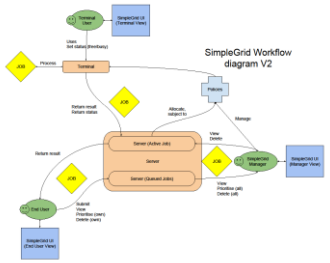


Figure 3: Work Flow Diagram, amended following new information received during the contextual enquiry

Converting the WAAD into system requirements

As can be seen from the first WAAD (Figure 2), separation and grouping by interface provides a certain perspective on the software, but is limited in terms of visualising parallels, and if developed into a system requirements diagram would contain duplication and cause confusion. We executed an abridged method of requirements extraction, which involved building a visualised task structure model with requirements and design note annotations from the WANs.

Noticing the commonalities of some tasks in the two interfaces suggested an alternative grouping was available, grouped by view and context. Time was spent re-working the WANs into a new order, to remove duplication, with the consideration of requirements and basic views/interface designs in mind. We were able to create groups of tasks which could then easily be divided into subtasks without duplication. We then labelled the task groups as three "views" with associated requirements and design ideas, as shown in Figure 4. We agreed that this model provides a clear task overview, and can be used to directly start sketching design ideas.

As the user groups and usage scenarios of the project are not yet clear, we did not explore task interaction models.

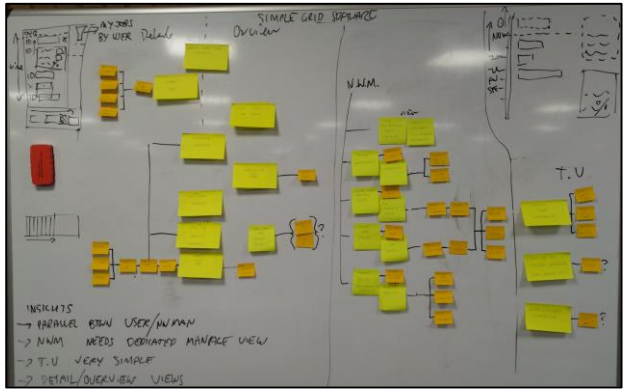


Figure 4: The created hybrid task/requirements diagram and its digital counterpart. Created following reorganization and development of the WAAD. This diagram clearly shows tasks, arranged by view and user, with minimal duplication. This model gives us a clear insight into the various views and controls that must be designed for full control of SimpleGrid across all interfaces.

Major insights gained

The final diagram which resulted from our enquiry and analysis highlighted four major insights:

1. There are many parallels between the EU and NM views, so a single view can be designed
2. The NM will need a dedicated management task panel
3. Views will be split into "detailed" and "overview" perspectives
4. The TU interface will be extremely simple and may be different to other views

Interaction with client

Email communication was established early on to evaluate the responsivity and expectations of the client.

Our first contact was a brief, productive meeting with Ian Miguel, who confirmed his role in the project and advised us on how to contact SimpleHelp.

The introductory conversation and interview with SimpleHelp were both conducted over Skype. Given that the customer is based in Edinburgh and that travel takes 1-2 hours, this was deemed more efficient use of time than a face to face conversation, given the 30 minutes and 1 hour required for the introduction and interview respectively.

Regarding the observation which was considered, the SimpleHelp team recommended we contact Professor Steve Linton of the School of Computer Science, St Andrews University as one a potential supercomputer user. The team personally consulted the professor and he offered to link us to managers of the supercomputer in the School of Physics. Following a short discussion with one of the supercomputer managers, the team concluded that there were not as many similarities in interface as anticipated, with much of the software being inaccessible and automated, and therefore decided not to conduct the direct observation.

References

Hartson, R., & Pyla, P. S. (2012). *The UX book, process and guidelines for ensuring a quality user experience*. *Morgan Kaufmann* (Vol. 37). https://doi.org/10.1145/2347696.2347722

CS5042 – P2
SimpleGrid

Iain Carson
University of St Andrews
ic48@st-andrews.ac.uk

Adamu Adamu Habu
University of St Andrews
aah5@st-andrews.ac.uk

Lan Liu
University of St Andrews
ll78@st-andrews.ac.uk

Scoping

SimpleGrid (SG) is a brand new piece of software still in conceptual stages. As such it has no current user interface, nor a fixed task flow or processes.

Work conducted on the workflow diagram and during the contextual enquiry highlighted three types of SG users; the Terminal User (TU), End User (EU) and Network Manager (NM). This leads to consideration of three separate views, or user interfaces.

The greatest challenges in the creation of SG's interfaces lie in presenting accessible supercomputer capabilities and visualization to inexperienced users; something which has, to our knowledge, not been done before.

Interfaces

The NM interface must allow visualization and manipulation of the current SG network status and job queue, as well as providing controls for the creation and modification of system policies to individual user and resource granularity. NMs would typically be IT literate but unlikely to be familiar with supercomputers.

The EU interface will visualize the network status and job queue in a fashion similar to the NM, but with interactions limited to the individual logged-in. Additional controls will allow the EU to upload jobs to the SG network for processing. EUs may be of a wide range of backgrounds and as such the interface must be able to accommodate them.

Terminal Users will be presented with a passive interface which will allow them basic control of how SG utilizes their terminal's resources.

SG targets current SimpleHelp customers, who may benefit from additional task processing capabilities beyond those that their IT system currently supports.

The group has considered all interfaces and views, and endeavored to offer "traditional" and "innovative" solutions within a coherent paradigm applicable to SimpleHelp's target customer base.

End User Interface

The EU interface starts the "chain" of information flow in SG, with the submission of jobs:

Submission of new jobs

The job submission interface must be simple to use and provide basic feedback to the user as to whether the job was successfully submitted, without being convoluted. As discovered during the contextual enquiry, EUs may be typical employees hoping to perform common tasks such as payroll calculation or video encoding, or more experienced mathematicians submitting algorithms and file processes. The process must be intuitive enough for simple submissions yet flexible enough for advanced operation. Once a job has been submitted it will have any relevant policies applied and be added to the SG job queue.

Visualization of the job queue

The job queue view is shared between the EU and NM, with the subtle difference that EUs can only manipulate and see details for jobs they themselves have submitted. We had to find a way to communicate details and an overview of jobs in a similar way to a print queue.

The traditional solution of a text-based print queue would likely suffice for this project, however the contextual enquiry highlighted the benefits of having greater detail and visualization in the job queue, as such we aimed to develop creative ideas which would communicate additional information and allow intuitive manipulation of items in the queue.

Network Manager Interface

The NM interface generated the greatest challenges in terms of innovation, and required the greatest attention to detail:

Control of system policies

Policies are made up of User Groups (a username and password; defines an EU), Resource Groups (collections of terminals) and Constraints (time and date periods). Policies are applied automatically to jobs as they are submitted, based on the User Group that submitted the job.

The NM must be able to define and edit policies, see which policies are active, which jobs have had policies applied and to manually apply constraints and resource groups associated with individual jobs.

The most obvious solution to this was to create a series of menus and selections in the style of a wizard. However, with further exploration and through sketching and brainstorming we developed clearer ways to achieve full functionality by incorporating drag and drop, and by introducing unique visual indicators.

Terminal User Interface

The TU interface is the simplest of the three interfaces. Whilst it has not yet been decided as to the level of control that TUs will be exposed to, we were tasked with creating an unobtrusive, intuitive interface that allows users with no knowledge of SG to free up their terminal resources if required.

Scenario sketches

Figure 1: Submission of a SimpleGrid job by an End User. The End-User is presented with basic information on the network utilization status, and decides to submit their job for processing. The job is processed and allocated to available computers. The result is returned and the End-User is notified of completion.

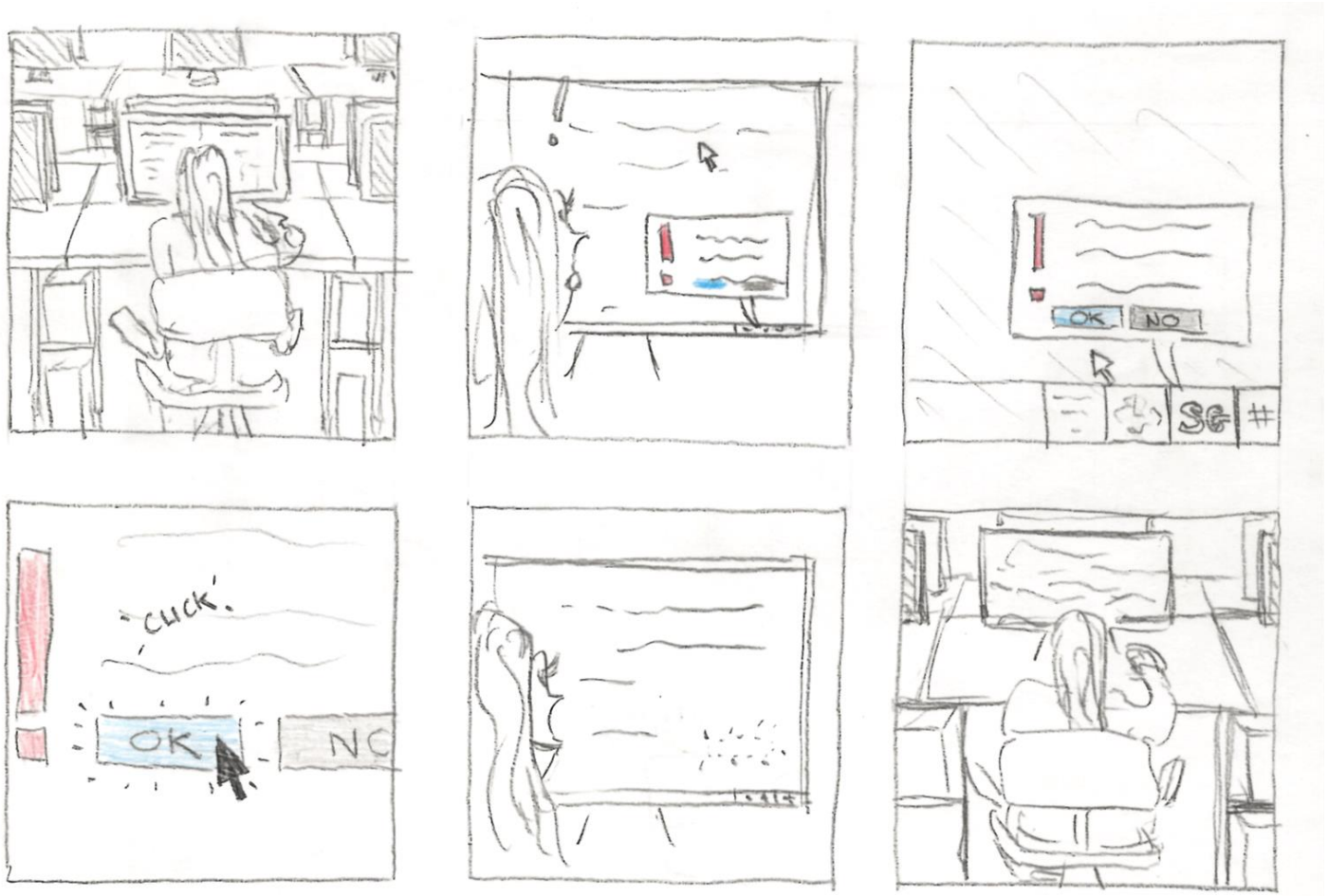


Figure 2: A Terminal-User is notified of the presence of SG, and allowed to make a decision on whether their terminal may be used for SG operations. The dialog is closed following the brief interaction. This is one potential scenario of interaction that the TU may experience. Other options may be more passive or active, feature a lesser or greater degree of control, or be purely informational.

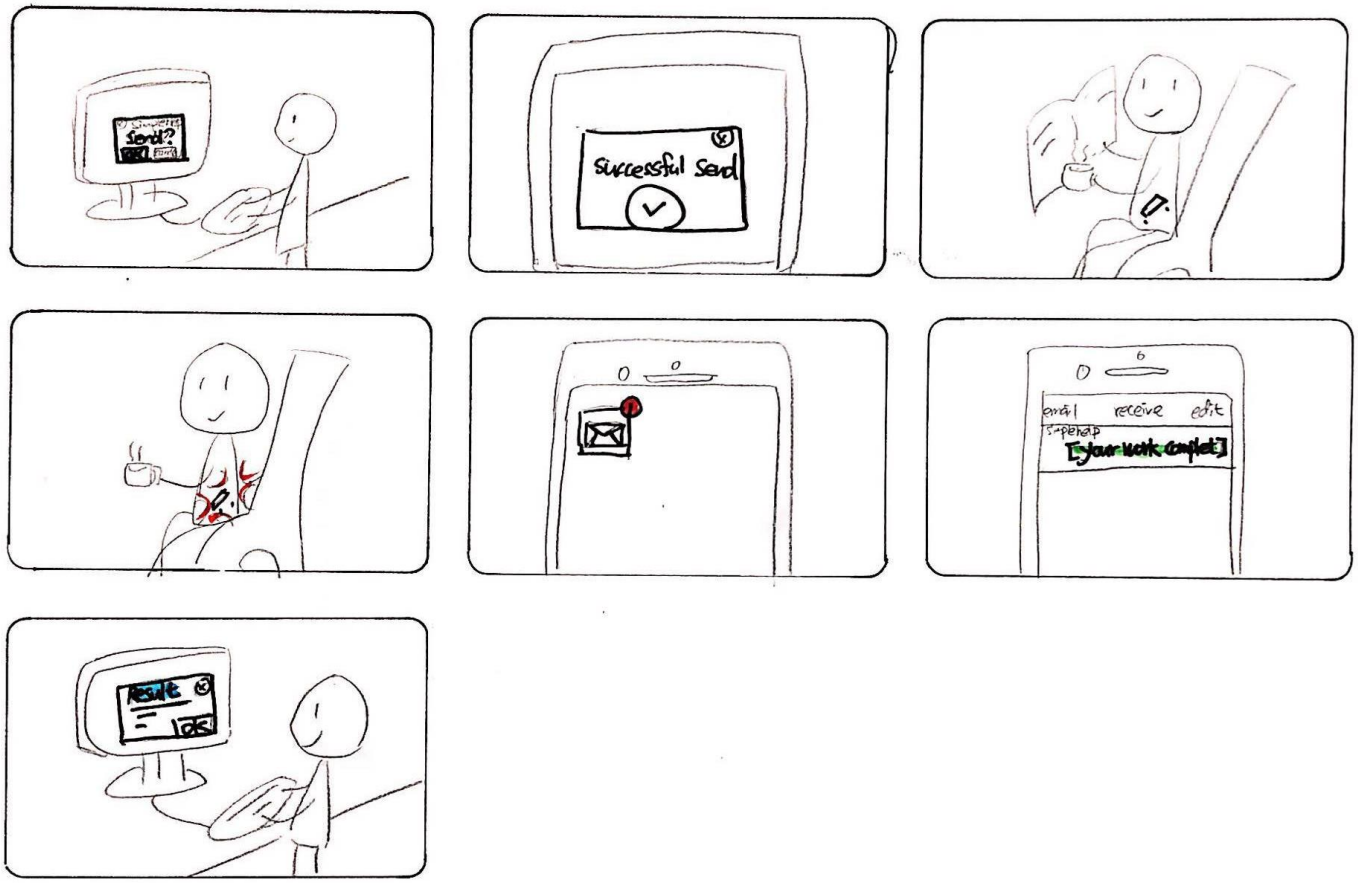


Figure 3: An End-User submits a job, leaves their terminal and is notified via email that their job has completed processing. They return to the terminal to evaluate the results of their job.

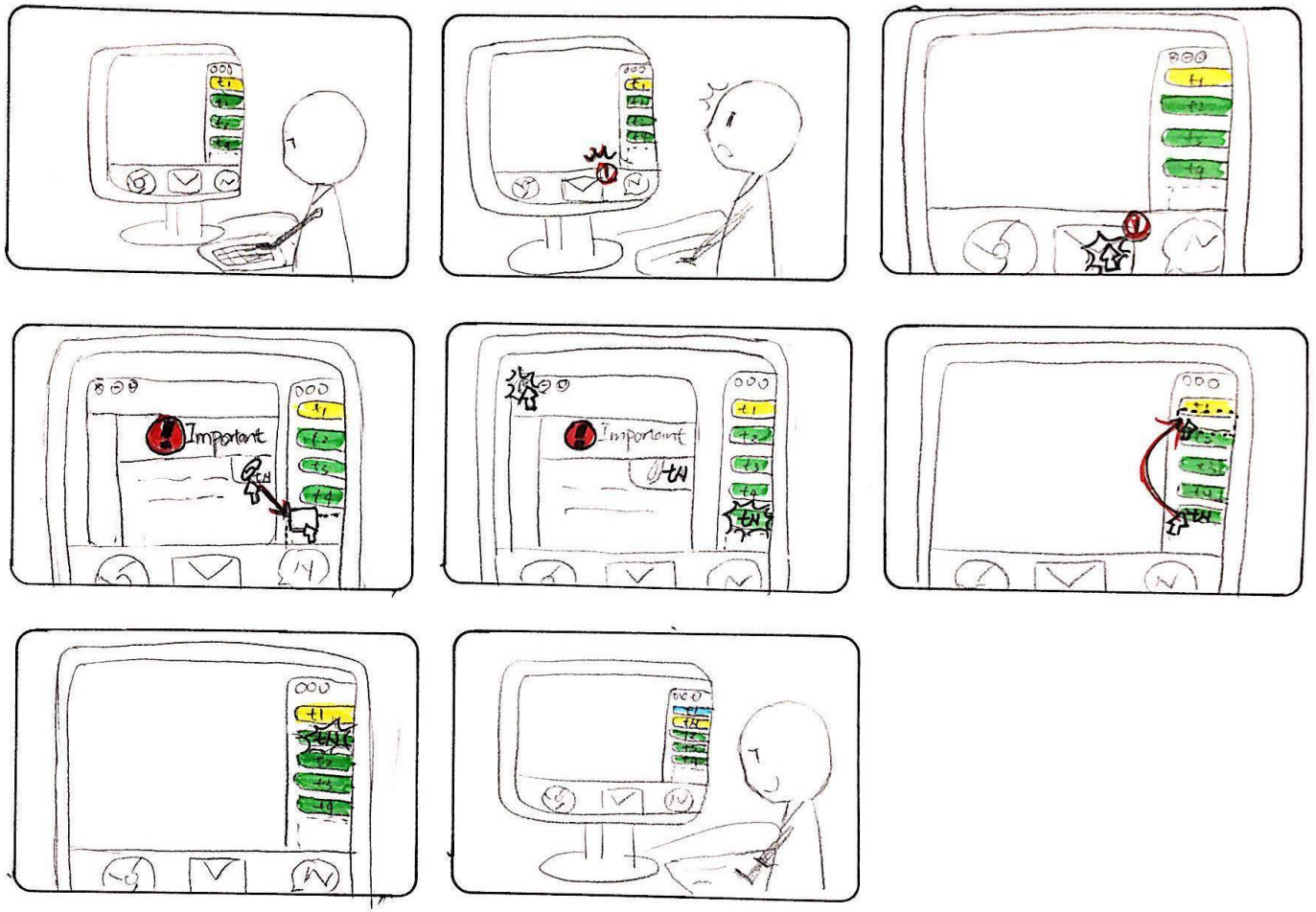


Figure 4: A Network Manager is notified via email that a job in the processing queue is urgent. They manually re-prioritize the job to ensure it is processed in time.

Solution Sketches

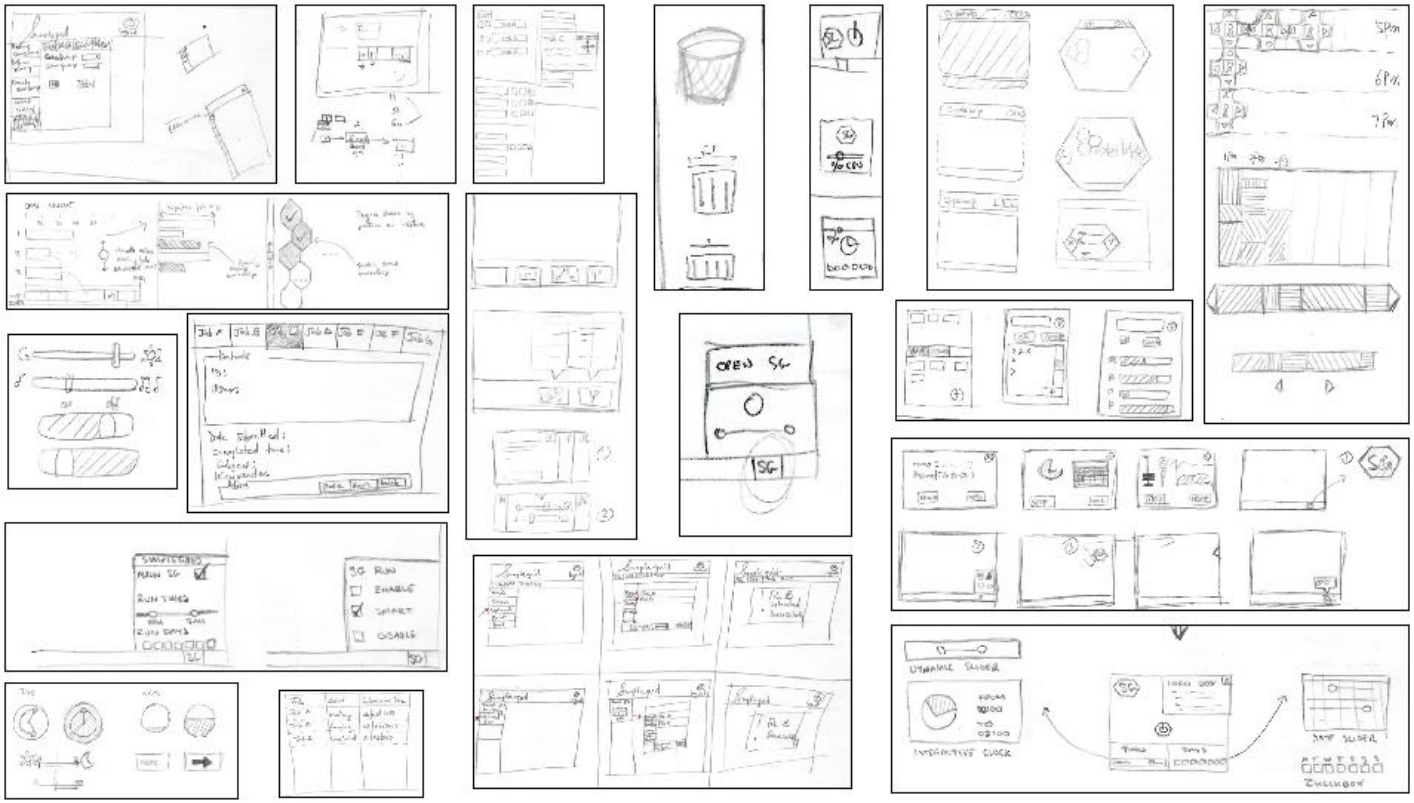


Figure 5: Illustration of solution sketching process for the more reserved of the two solutions proposed. Details overview, end-user, terminal-user and network-manager interface exploration.

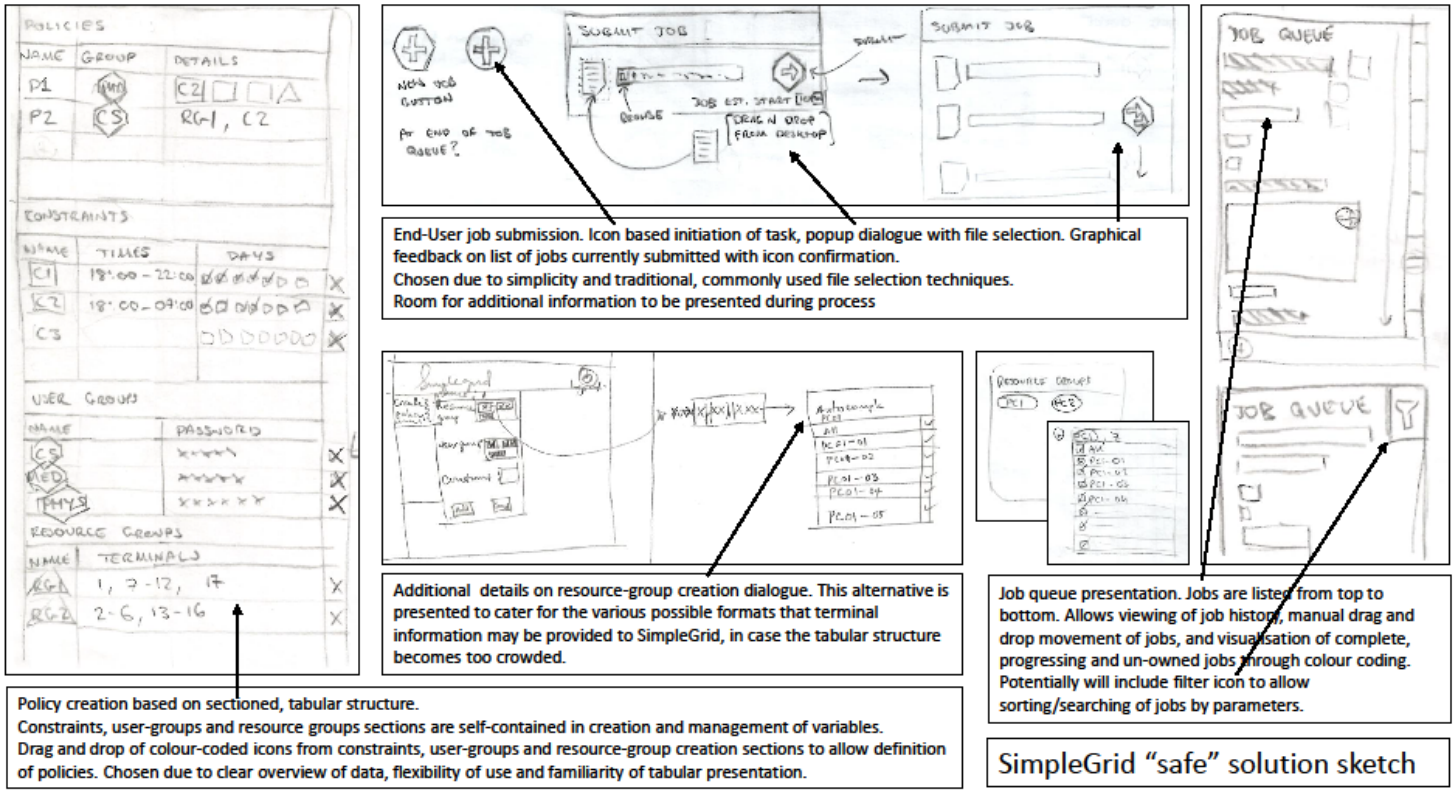


Figure 6: The SimpleGrid “Safe” solution sketch. This annotated illustration summarises many of our ideas and approaches regarding the SimpleGrid interface.

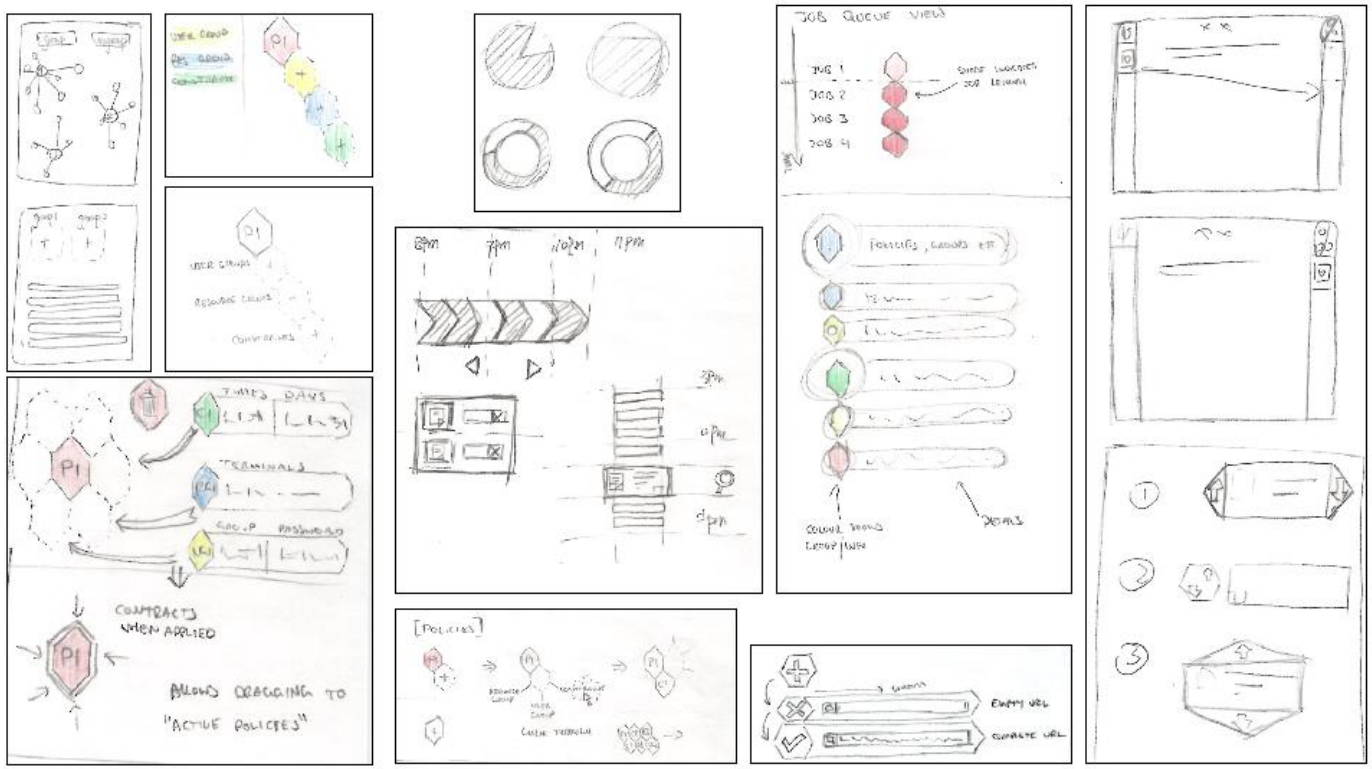


Figure 7: Our sketching process for the "crazy" solutions offered. Use of icons, symbols, controversial shapes and animation-dependent processes was encouraged to explore more abstract possibilities for interaction and information display.

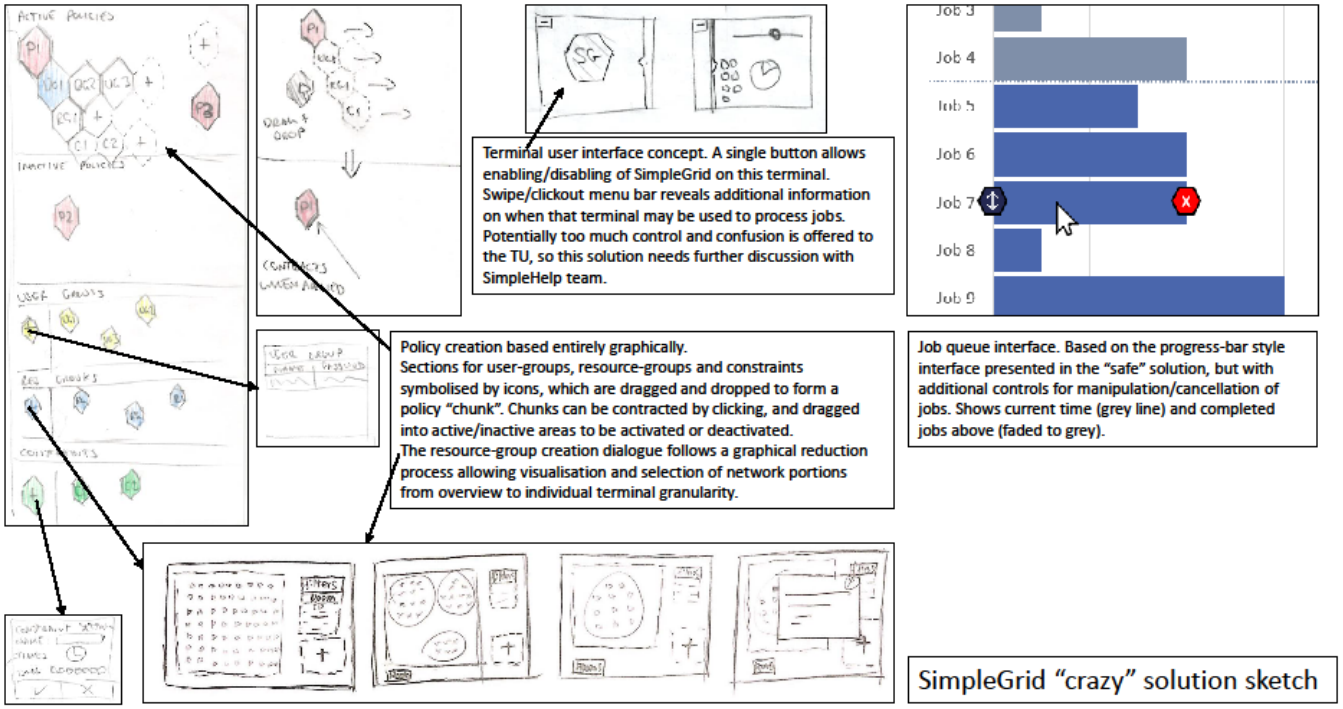
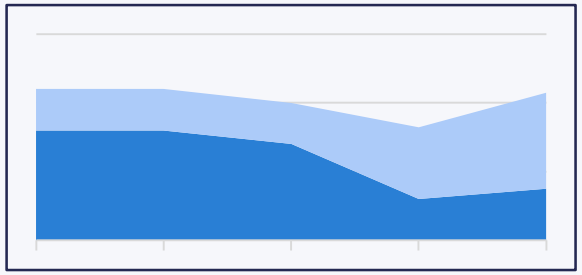
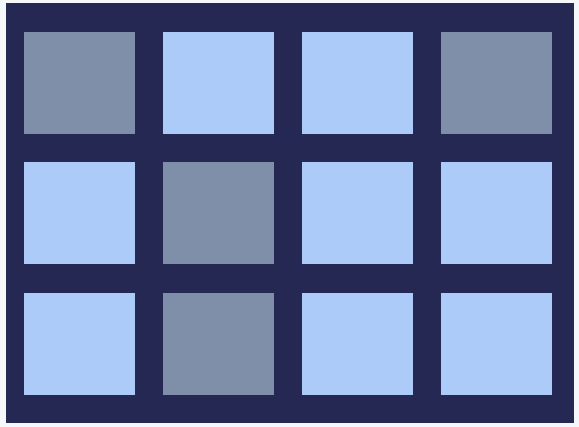


Figure 8: The SimpleGrid "Crazy" solution sketch. This annotated illustration highlights our more original and controversial ideas that evolved through group discussion of the SimpleGrid interface.

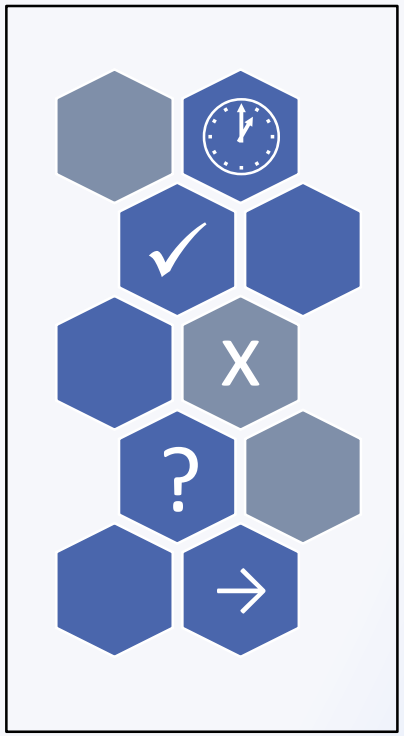
simplegrid

1
2
3
4
5
6

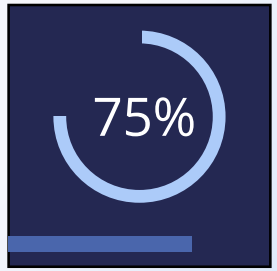


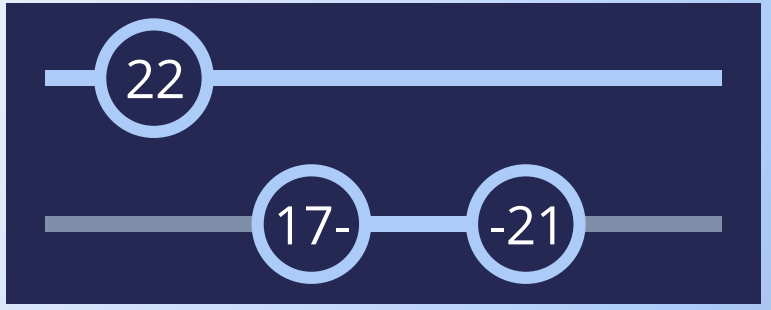


OK Cancel



Mo	Tu	We	Th	Fr	Sa	Su
	1	2	3	4	5	6
7	8	9	10	11	12	13
14	15	16	17	18	19	20
21	22	23	24	25	26	27
28	29	30				





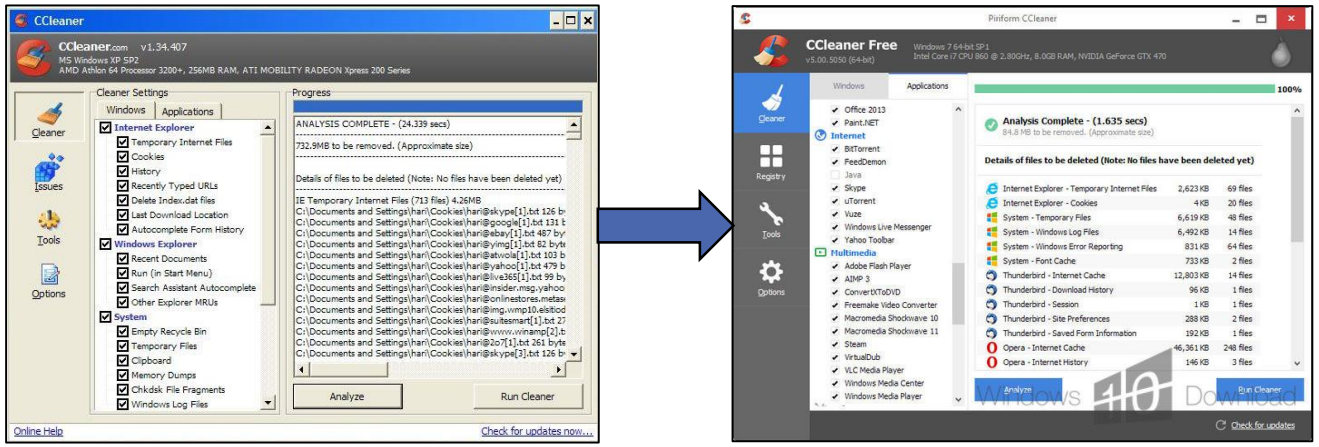
Moodboard and Material Gathering

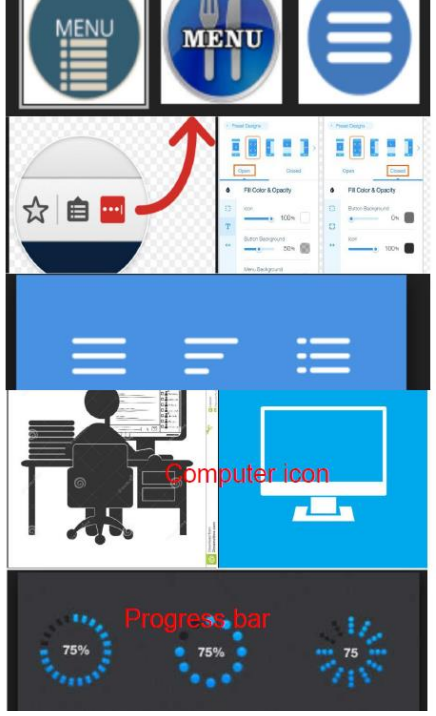
The moodboard presented is the result of combined group efforts to collect materials, discuss the style and mood of the intended application, and to create original content which accurately reflects and represents the desired mood.

Group members worked individually to collect and collate various sources which inspired or influenced the mood imagined, based on inspiration from the original SimpleHelp website:

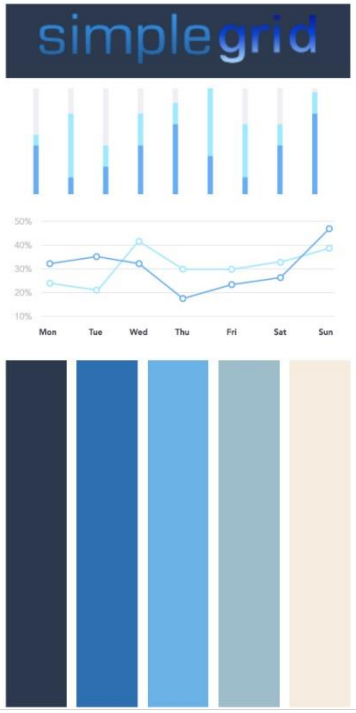
Some layouts and icons were inspired by SimpleHelp's current software, though given the age of the system an opportunity for modernisation was recognised.

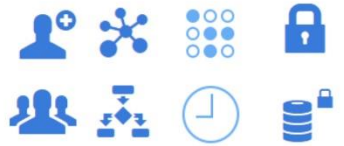
The current system very much resembles typical Windows/Linux applications from around ten years ago, and we wanted to modernise SimpleGrid to fit in with current design paradigms, in a fashion similar to that demonstrated by the popular software CCleaner:



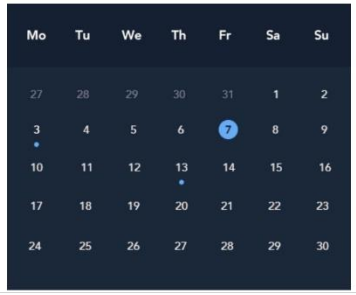


Materials gathering



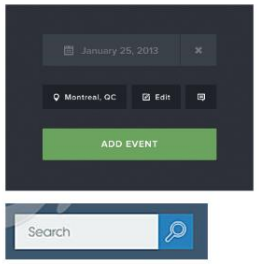


simple
tiled
modern

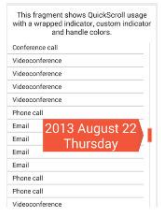


USR-12	• 67% • No issues
USR-114	• 00% • No issues
<TEST>	• 00% • No issues

button



option design

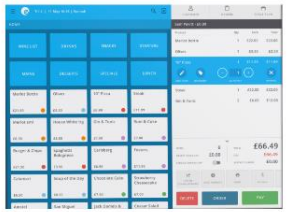


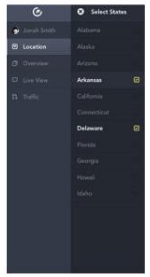
icon

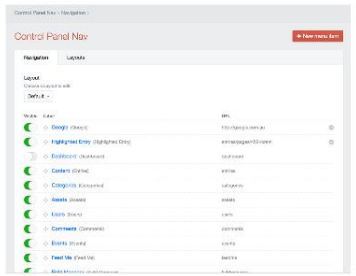
color

time & RAM

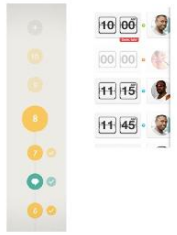
menu



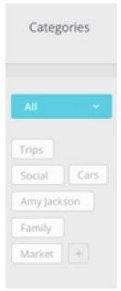




user list



protocol select

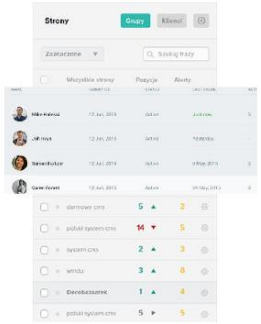


user overview

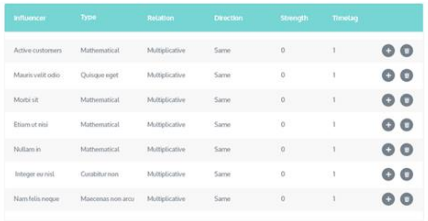
color

icon

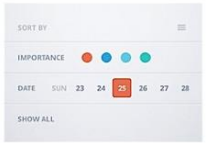
task overview



menu



user selection



CS5042 – P3
SimpleGrid

Iain Carson
University of St Andrews
ic48@st-andrews.ac.uk

Adamu Adamu Habu
University of St Andrews
aah5@st-andrews.ac.uk

Lan Liu
University of St Andrews
ll78@st-andrews.ac.uk

Concept Statement

SimpleHelp is a company that provides remote support software. Their flagship product is SimpleHelp, software and software infrastructure that allows clients to take control of their customer's desktop (see https://simple-help.com/simplehelp). SimpleHelp is interested in expanding their product to enable clients to set up private computer grids that perform computation-intensive tasks by using machines that are underutilized at any particular moment.

This approach promises significant financial and CO2 savings; a job administrator can use their existing base of computers as a data center/High Performance Computer without having to purchase, install or rent expensive infrastructure, and leverage computing power that is already installed and maintained for another purpose (employee computers).

This new functionality of the software will require a new interface that enables certain kind of users (job administrators, grid managers) to submit, schedule, monitor, and receive results from the computations carried out on the grid. The quality of the interface is crucial: it should enable sophisticated configuration of a potentially complex distributed software-hardware system while making the process and its monitoring simple enough to make the configuration of new tasks and grids worthwhile.

The creation of the jobs themselves is out of the scope of the project, since the jobs will usually be built using standard programming tools such as IDEs. The team will focus on extracting meaningful requirements and proposing one or more graphical user interfaces that support job administrator activities.

Interface Goals

SimpleGrid (SG) software relies on complex algorithms to divide and distribute computationally expensive tasks for calculation. The nature of the algorithm is beyond the scope of this project, however it will fundamentally require creation and prioritization of a "job queue", with jobs executed sequentially as per the queue.

A Network Manager (NM) will be responsible for the setting of system parameters which affect the prioritization mechanism and the systems over which the jobs are distributed. The contextual enquiry conducted at the beginning of the project highlighted that this should to some degree be automated through the use of "Policies", yet allow for individual job priorities to be overridden if necessary. NMs may also have the ability to view SG resources are currently being used, at an aggregate or granular level.

The End User (EU) is a software user belonging to a UG, able to submit jobs to the system, view progress of their jobs and their position within the queue. The possibility of deleting or reprioritizing their jobs was considered as well, though was not implemented in the final prototypes as there are still multiple options to choose from. Higher fidelity prototypes with user feedback will allow narrowing down to a particular mechanism for reprioritization of jobs.

Finally, a Terminal User (TU), who may not be aware of SimpleGrid and its purpose, may have some degree of control over whether SG may run on their terminal. The exact requirements for this interface are not fully specified, as its functionality will depend heavily on the final design of the job allocation algorithm. In any case, information should be provided to allow the TU to make a sensible decision on any presented controls, for example if performance of his terminal is adversely affected by SimpleGrid, he should be made aware why, and also be made aware of the consequences of taking action to disable SimpleGrid. Several options are presented for review and consideration by the client.

Between the NM, EU and TU, three different views or interfaces, each with different requirements, and featuring options still yet to be decided on, must be presented for completion of the SimpleGrid user experience.

The designs presented in priority may be considered more "out of the box" solutions, however they are intended to be as functional and usable (and perhaps even more intuitive) than their "conservative" counterparts. Features such as the graphical, bar-based job queue, although unconventional, were felt to have such strong advantages over traditional representation that they have been utilized in both designs.

The prototype interfaces presented are not intended to cover all features requested in all interfaces, as this would be too large and complex for this project. Instead, they present a framework around which the final software workflow may be built, and details into some of the more difficult or complex tasks, which have been simplified to allow completion through guided, visual paths.

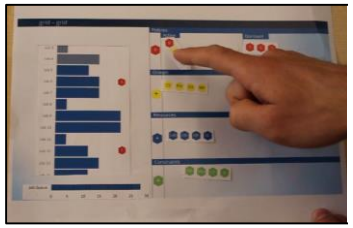
Network Manager Interface

Error! Reference source not found. shows the design of the Network Manager interface.

The unique interactive nature of the interface made it unsuitable for prototyping using myBalsamiq or similar, as the functionality would not be well represented when limited to single clicks or drop-downs.

Paper Prototype

The Network Manager's interface was therefore implemented as a fully interactive paper prototype, pictured below:



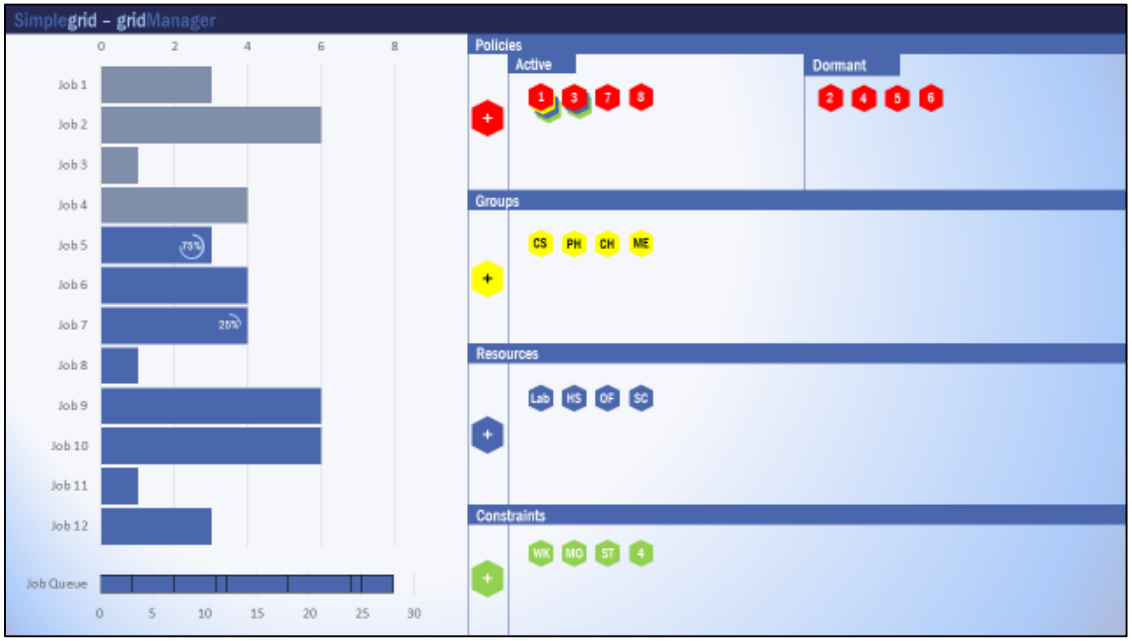


Figure 1: The Network manager design and prototype. This out-of-the box solution is highly graphical and relies on drag and drop, visual encoding and symbols to guide the user through the various features available.

Designs

Network manager

Figure 1 shows the NM interface; the most complex of the interfaces presented, as it requires interaction with the job queue through creation of policies, groups, resources and constraints, key requirements as set out in the contextual enquiry. Full visibility of the job queue and all submitted jobs is provided at a glance, and upon clicking a job the NM can view details of individual jobs in a popup. This gives them a system-wide overview at a first glance (viewing the system at an aggregate level), whilst enabling zooming into finer details if required. Policies and their associated resources

Scrolling of the job queue, and creation of new policies and components through drag and drop were simply unachievable through the provided digital prototyping tools, so to do the interface justice a different mechanism was required. As such, the NM interface was conceived as a "Wizard of Oz" paper prototype. A 5-minute video recording of the major features of the prototype can be viewed here:
NM Paper Prototype Video

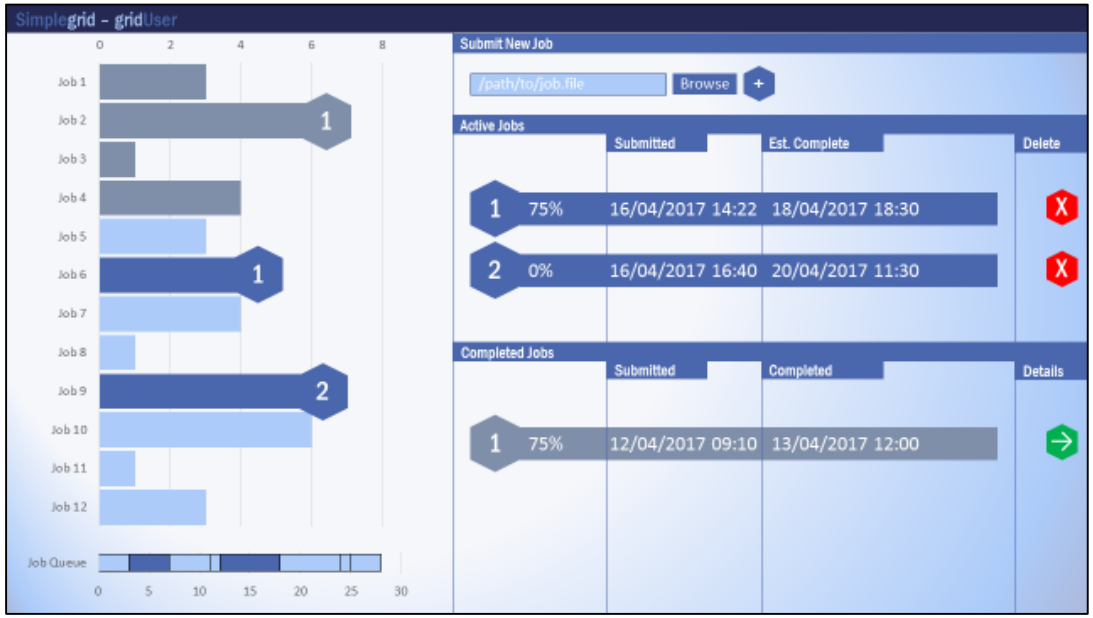


Figure 2: The End User interface. This closely resembles the Network Manager interface on the left, with additional symbolism and coloring allowing differentiation between the user's jobs and other jobs, and labelling showing the corresponding details to their jobs on the right hand side.

End User

The requirements highlight that the EU interface should allow new jobs to be created, viewing of current jobs in the queue and reviewing of active and completed jobs belonging to that user.

As such, the EU interface features a job queue overview similar to that of the NM, but tailored to that user's submitted jobs. The controls on the right hand side from top to bottom allow creation and submission of a new job, viewing of the progress and details of active jobs (as well as deleting by clicking the red cross, or re-prioritising through drag-drop), and viewing of the results of completed jobs (with the green arrow).

The greatest disadvantage of both the EU and NM designs in this context is the quantity of information that may be displayed in a single screen. As the number of items increases (be it jobs in the queue, active policies or resources, or even completed jobs) the interface may become unmanageable. Filtering, sorting and perhaps searching are therefore required, and this would likely take the form of a filter menu present at the top-right of the job queue.

Sketching Derivatives

The UIs presented derive directly from sketch ideas submitted in the second assessment. A clear example of this is the Network Manager Control "conservative" interface, sketched in Figure 4 and rendered in high-fidelity in Figure 6.

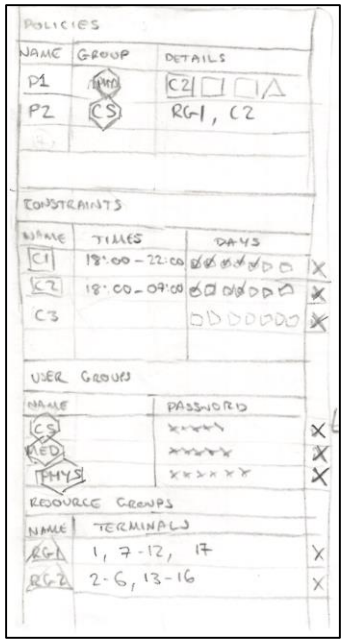


Figure 4: NM Policy Creation tool sketch

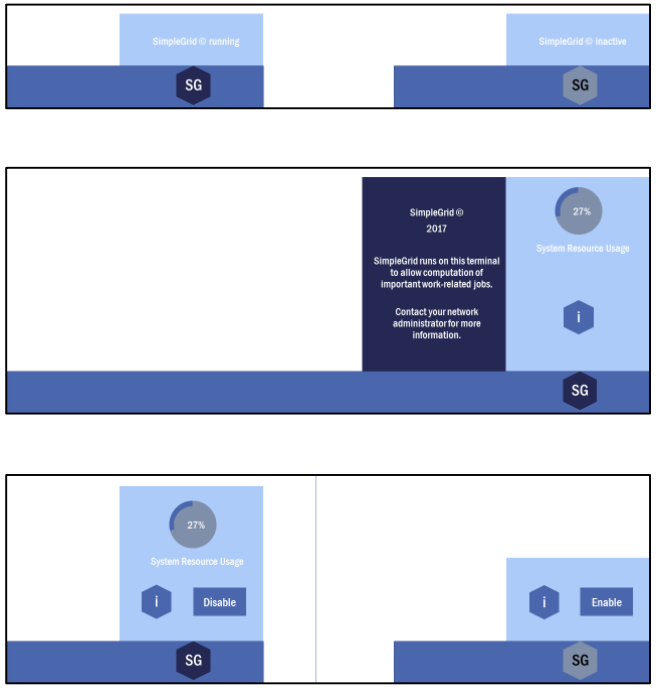


Figure 3: Terminal User Interface designs. Three designs are presented.

Terminal User

Figure 3 shows three alternative designs for the Terminal User. Given the uncertain nature of SimpleGrid's sorting algorithm and its robustness against having TUs making decisions, differing levels of interactivity are provided.

In Figure 3, from top to bottom:

1. The TU is presented with basic information informing them whether SimpleGrid is running or not. No further details are provided, yet no controls are provided either. Removing all control from a TU may be desirable in contexts where terminals are limited and can be micromanaged by the NM.
2. The TU is presented with information regarding the impact of SG on their terminal. They may click on the "i" button to find out more information about SimpleGrid. This interface is demonstrated at the end of the aforementioned video as an interactive digital prototype for presentation.
3. The TU is presented with both information on their system resource usage (similar to option 2) but also the ability to enable or disable SimpleGrid.

The presence of SG as a tray icon was chosen as for the most part, TUs will take no action whether the software is running or not. For minimal impact on their working environment it must therefore remain subtle.

In all options however, the icon will change color depending on whether a job is executing on that terminal.

Network Manager Controls alternatives

Policies		
Name	Group	Details
P1	CS	C2,RG1
....	....	.....
Constraints		
Name	Time	Days
C1	18:00-22:00	Mon,Wen...
....	....	.....
User Groups		
Name	Password	
CS	St-CS12345	
CS2	St-CS23940	
....	.....	
Resource Groups		
Name	Terminals	
RG1	1,2-6,11-35	
RG2	20,30,7-8	
.....	.....	

Figure 6: The conservative approach to creation of policies, constraints, user groups and resource groups. These items are defined through text input, and assigned to their policies through autocompleted-text input again.

The textual nature of this interface was shunned as it can become overwhelmingly complicated when multiple text fields with no color differentiation are presented.

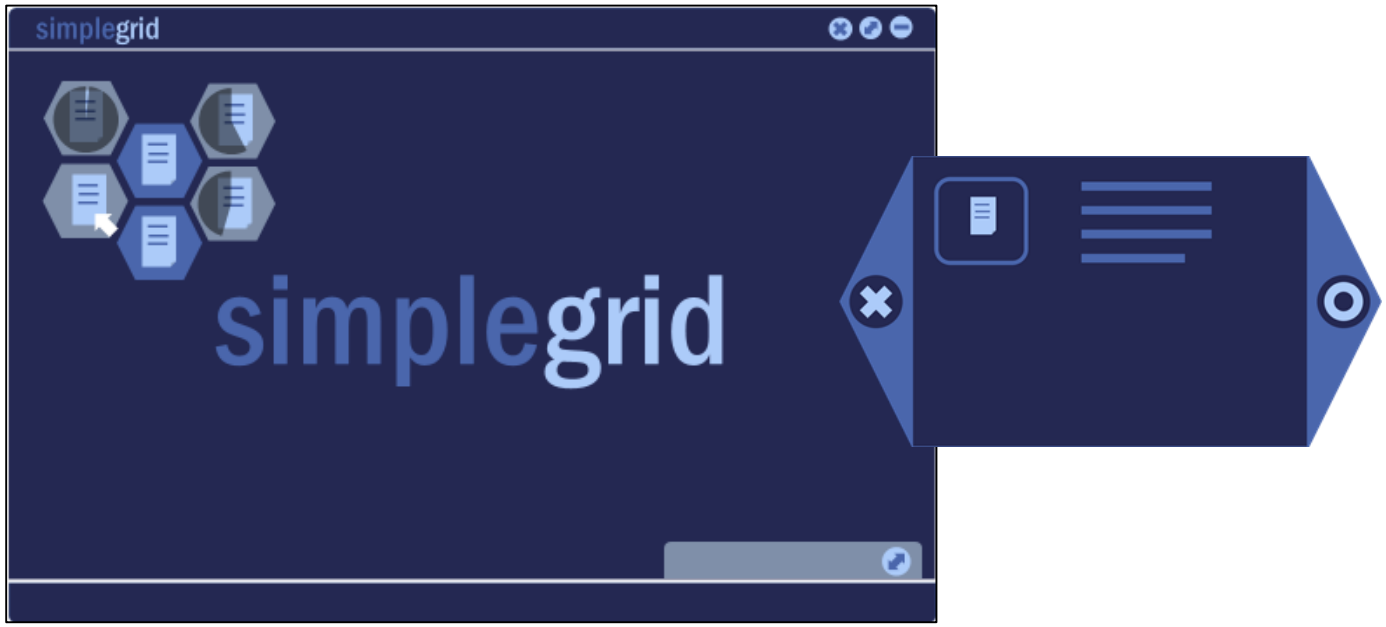


Figure 5: Aesthetic design alternatives. The dark background, hexagonal icons and popups and "chunky" graphics were a strongly considered aesthetic for SG. However, this design was not implemented in the end as it was considered too alternative for the business environment and anticipated customer group, who are likely to prefer designs more conservative in nature.

Design Alternatives and Evolution

Network Manager Controls

Figure 6 shows a less graphical version of the NM policy creation tool. This alternative was designed early on as one of the most obvious solution for the entering of requirement information associated with policies. However, it was quickly realized that because of the heavy amount of text, this interface may become overwhelming before very long.

Aesthetic Design

The moodboard presented a clear yet flexible color and shape scheme for SG. Various alternatives for the nature of the overall interface and the popups are presented in Figure 6.

Professionalism and consideration of the design aesthetic required a lot of designs to be reworked many times. Although a long process, it was agreed that the professionalism of our presentation will, in the mind of the customer, reflect the degree of effort put into the project. For maximum impact and the highest chance of design acceptance, high professionalism was required.

Resource Group Creation Sketches

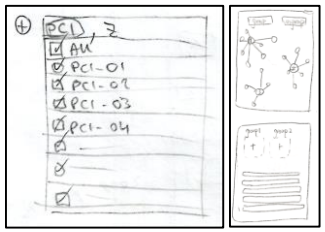


Figure 9: Consideration of multiple designs for resource group creation.

Resource group creation was one of the most challenging aspects of the Network Manager interface, as it requires a visually engaging and intuitive way to filter from potentially thousands of terminals into a select group.

This required the most sketching and discussion regarding details.

One of the most promising final designs is presented in the video of the paper prototype.

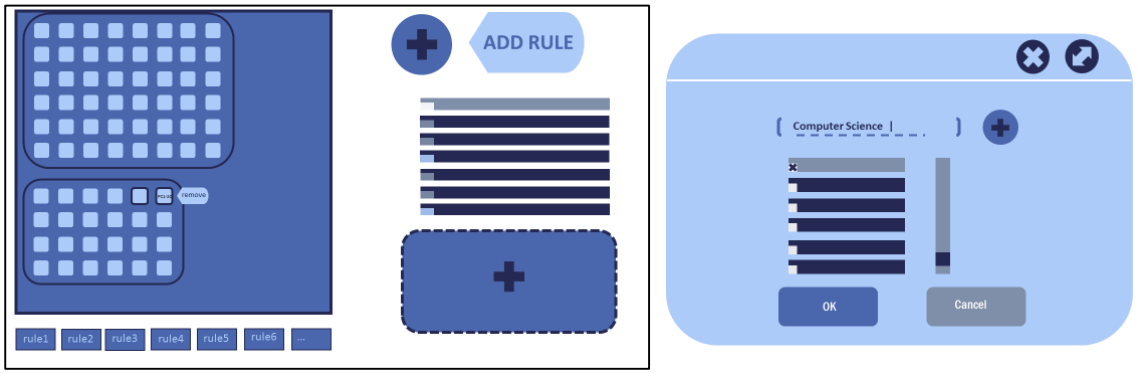


Figure 7: Resource Group creation and popup alternatives.

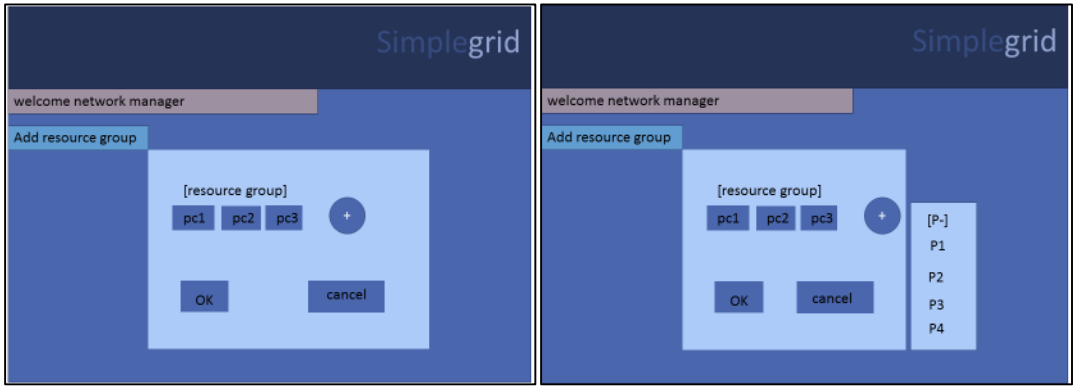


Figure 8: Network Manager Resource Group creation alternatives – conservative version.

Resource Group Creation alternatives

Figure 7 shows two high fidelity designs for the creation of Resource Groups. The principle is that a user may apply "rules" to the full set of terminals to perform filtering on the terminals assigned to the group. Creation of rules is performed using a popup which appears when the "+" button is clicked. Graphical feedback in the form of named squares representing terminals tells the user what has been assigned to each selection. This screen and associated popup were eventually dropped from the final design as it would have been the only policy component creation tool which featured more than one popup. For consistency, a simpler, less visually complex design was therefore chosen.

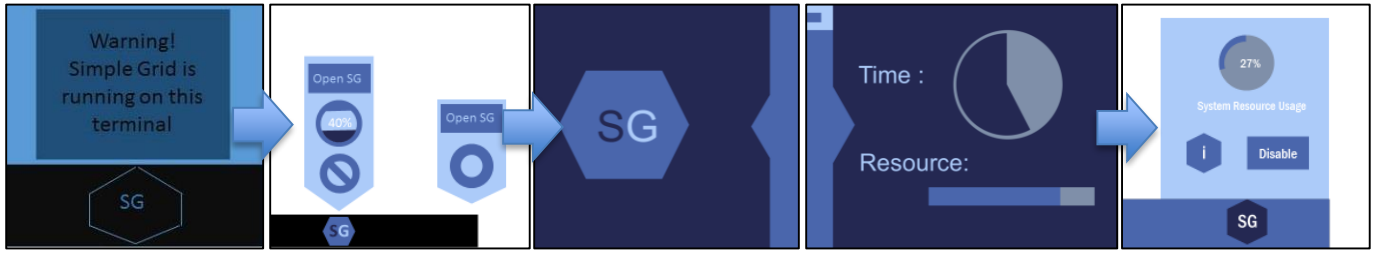


Figure 10: Design Evolution. Various interface components were designed together, through discussion and combination of thoughts. For example, above, a Terminal User interface option (right) was generated through combination of an information-based (left) and graphical (center) design. Fidelity and professionalism were deliberately improved on with each iteration.

Design Limitations

The various designs presented have developed over a period of discussion and review, and as such represent the best efforts of the group in light of the information generated through contextual enquiry. However, as the contextual enquiry was conducted very early on in the process, and communication with the client has since been relatively limited, there designers admit gaps in the interface due to lack of intricate system knowledge. Further limitations are generated when it is considered that the algorithm and system does not yet exist.

An attempt to circumvent these limitations was made by consideration of multiple alternatives, and by deliberately leaving extreme details out of the prototype. An ideal next-stage would be a second contextual enquiry, with specific focus on allowing the user to interact with the paper prototype and observing closely their instinctive workflow and response to the interface's feedback.

Paying particular attention to what information the user expects to see at each stage in the interaction process will allow iterative design through more productive resource creation interfaces and job summary screens.

simplegrid

Policy creation tool

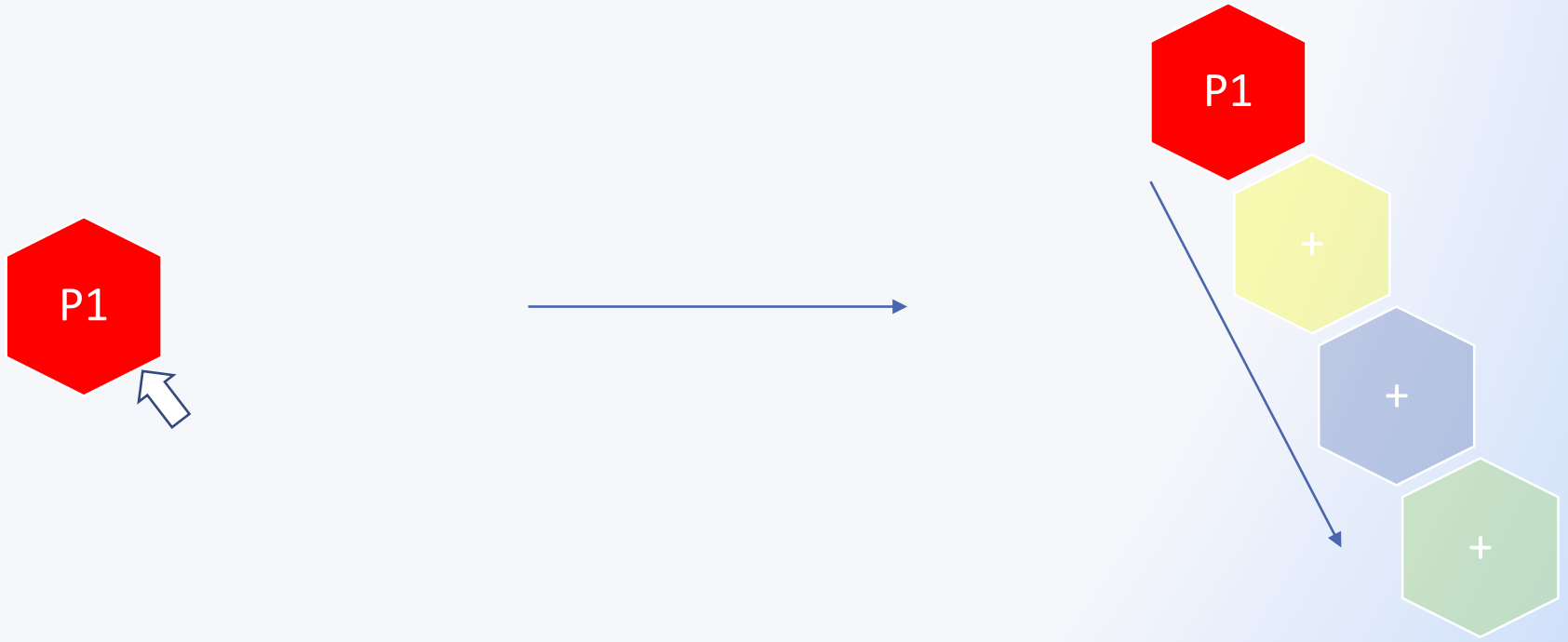
Policies

- Sets of rules
- Automatically applied
- Visible
- Affect job execution:
  - Order
  - Terminals
- Set by administrator only

Policies consist of

- Constraints
  - Time, date
- User Groups
  - “Submitted by”
- Resource Groups
  - Selection of terminals

Policy Creation Tool



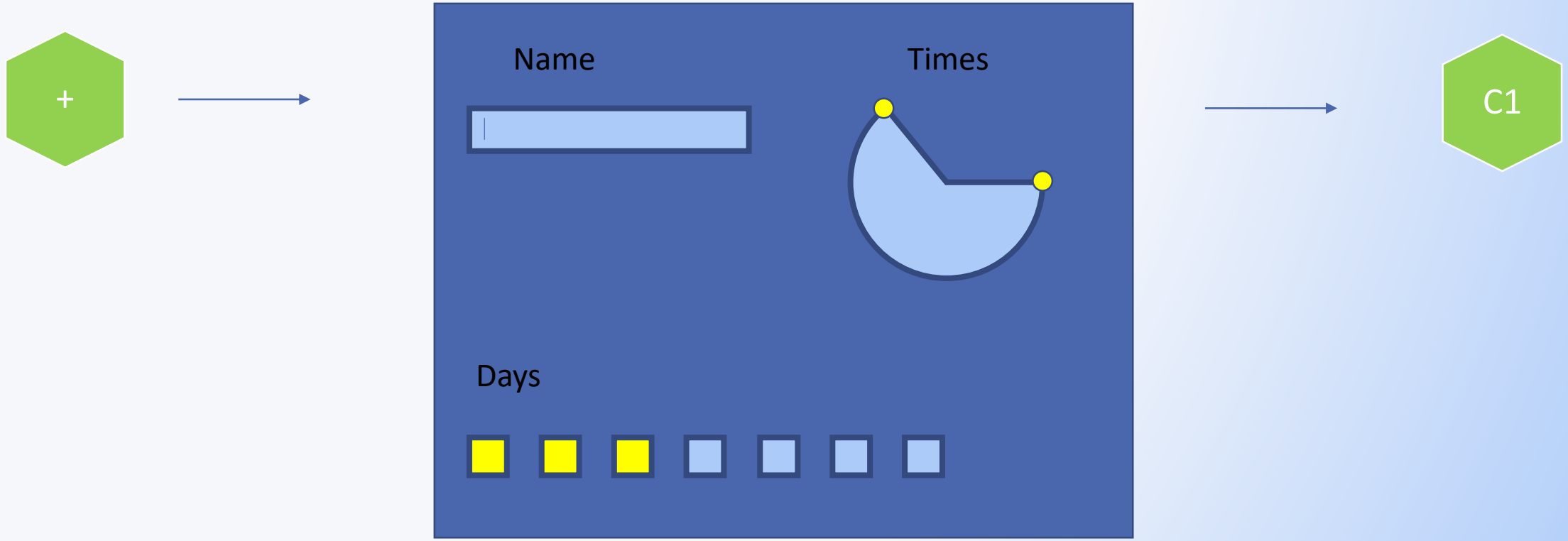
Item creation boxes

User Groups

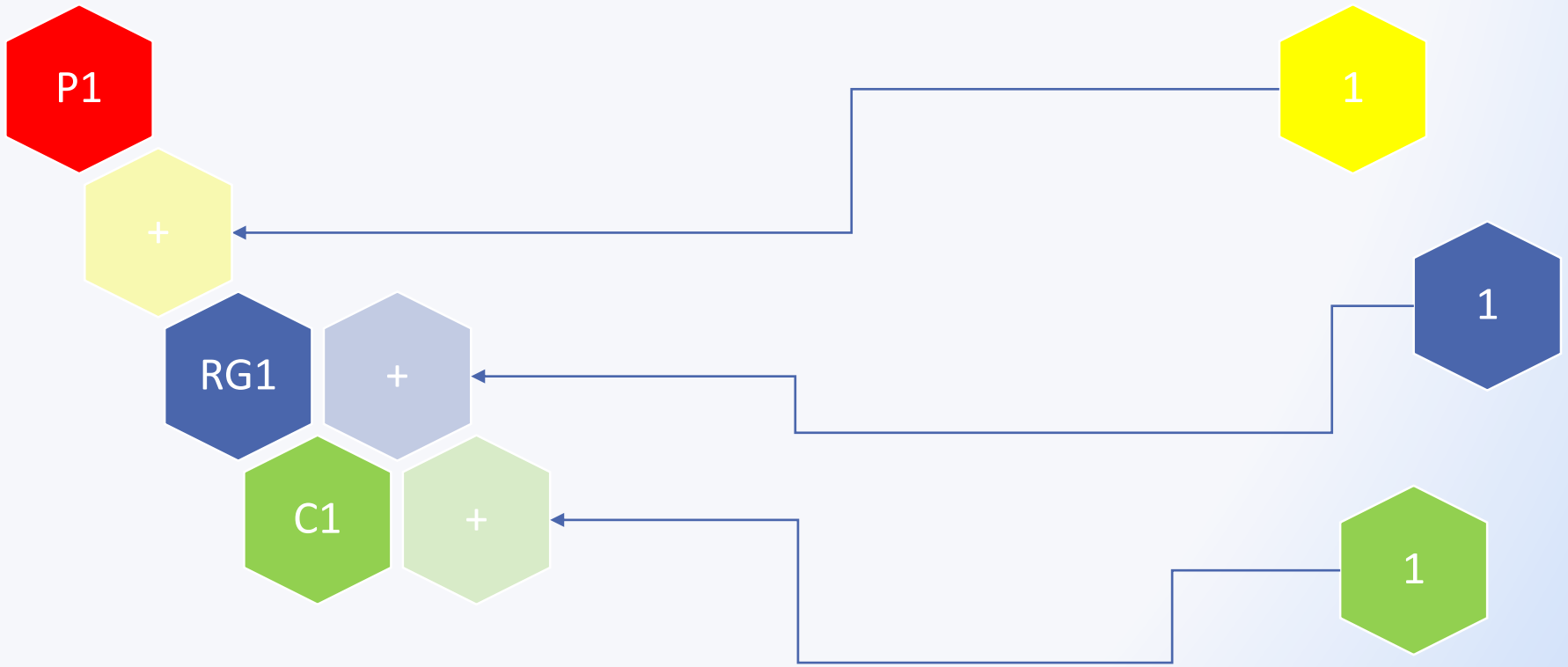
Resources

Constraints

Constraint creation



Creating a policy



Completed policy