# Grid Computing – An empirical perspective

**Abstract**

The intention of this paper is to critically evaluate the performance of the empirical perspective and *tkeden* toolkit with relevance to the creation of educational, decision support and simulation tools in the field of grid computing. A suitable model will be provided to demonstrate the major features of this philosophy and highlight several of its benefits as well as short comings.

## 1 Introduction

Grid computing is fast emerging as a realistic and probable contender for the way in which we undertake bulk processing jobs in the future. This discipline, aiming to tie together the task creator and a capable, remote resource provider is still a new area of research with an ever increasing world-wide base of interest. It is the intention of this project to carry out feasibility and educational modelling studies upon this domain of grid computing to assess both the extent to which empirical principles can help in the understanding of this vast subject area, and also the potential pit-falls which this field of computational science may face in its future.

### 1.1 Grid Computing Overview

In this grid model we will assume a generic version of the Open Source Grid Architecture (OGSA) model with three main tiers. An application (user) level, a grid middleware tier (resource brokering) and a resource tier (the grid 'fabric') (OGSA, 2000). One of the primary functions and difficulties of current grid implementations is in ensuring a standard level of service. As such many approaches implement a Quality of Service (QoS) agreement between users and brokers to determine the minimum level of service which can be expected.

## 3 Modelling Aim

The purpose of the modelling study is to assess the strengths and limitations of empirical principles as applied to the specific domain of grid computing. The study will apply standard concepts including agency, abstraction, artefacts and animation to help assess the suitability of these theories and the possibility of new concepts of problem cognition. In order to guide the study in such a diverse domain it is necessary to define the likely system 'views'.

### 3.1 Decision Support

Decision support systems (DSS) are increasingly being used by both high level executives but also low level facilitators to assess the optimum (or at least the optimum as perceived by the individual observer) configuration for a system in order to achieve a specified objective (Beynon, Rasmequan & Russ, 2002). In the case of a grid computing resource centre it may be necessary to maintain the resource centre at maximum utilisation (capacity). It is important to understand the inherent objective differences between users of the system as made apparent by previous studies. Where the executive may be looking to process a job schedule which will allow for maximum profit, the engineer may be looking for maximum sustainable system utilisation.

In lieu of this it is necessary to accommodate both parties by providing an objective system capable of both roles and user view points, for example it may be necessary to show a statistics page for easy examination of the internal system state, figure 1 shows such a screen.

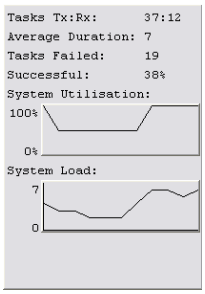

Figure 1 : system statistics

Designing a system capable of supporting these multiple view points and allowing an accurate overview of the system will help illustrate the effectiveness of empirical modelling at problem abstraction and ultimately 'domain moulding', whereby a specific abstracted domain may be applied to many different user view points through a single model architecture.

### 3.2 Educational Development

In addition to the decision support requirements the model should also allow for a simplified simulation of a generic grid computing environment to aid in educational practices, including the use of interactive, exploratory learning. In order to achieve this highly abstracted level of inference it will be necessary to simplify the system architecture and model using standard agentification techniques and LSD analysis. It is a vital part of explaining a model to an observer that the visualisation used to illustrate the process contains several artefacts to which the user can relate and thus form a positive and accurate mental model of the systems processes. This can, in part, be achieved though good UI design and clear segregation, linking and labelling of agents, using visual cues wherever appropriate.

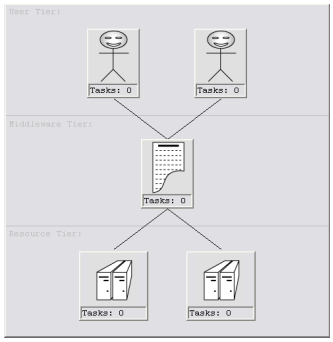

Figure 2 : visualised system overview

## 4 Empirical principles

Empirical practice mandates a method of abstraction away from the core 'real world' system which is trying to be modelled in order to create a finite set of controllable objects within an effectively closed world. This abstraction allows the system to be simplistically modelled whilst still allowing for a realistic exploration of the state space.

Abstraction can best be shown through the use of agentification techniques designed to find the core agents within a complex system, i.e. those objects within the system with the power to alter the state of the model. This agent orientated approach, applied to the grid computing paradigm will allow the modeller the determine the key agents for change within the system, thus allowing a reductionist approach to filter out all peripheral objects which do not significantly alter or affect the system state.

For the duration of this paper we will be taking a view that agents are defined as objects responsible for state change within the system (an empirical 'view 2' (Beynon, 2004) approach to agency). This is to help reduce the complexity of treating every object (such as network links, etc) as an individual agent whilst retaining the interest of a non-deterministic system with virtual agency in a closed world.

There are several immediate candidates for agents within the system (using the abstracted three tier approach as described earlier:

- Upper tier users
- Middleware broker
- Lower tier resource providers.

It is possible to prove these agencies by determining a causal link between the agents action and a state change within the system. In this case all three agents are capable of changing the state of the system, either through creation, distribution or manipulation of tasks. It may be useful here to include a notion of co-existence between objects and agents, whilst not giving objects a pivotal role within the model it is important to ensure they are included to enable an accurate model, and to provide the observer with valuable experiences from the system.

In the system in question we will assume a static set of agents using the M-agents principles to define an inter-relation existing between multiple dynamic parties each with more than a simple reactionary response.

In order to allow complex, concurrent multi-agent systems to be successfully experimented upon there is an important notion of anima-

tion of the state change. This allows monitoring of system state over time and reclassification of agents through experimentation.

One of the pivotal and revolutionary points in the use of empirical techniques is the recognition that the system modeller has a view point on the system, and thus an unconscious bias towards artefacts and agents which make sense in the context of their experiences. Once this situation is understood it is possible to study the effect this may have on the model, and thus how it should be amended to connect with the target observer, whomever that may be. In the case in question this may be achieved by attempting to stick to cultural norms, shared experiences and knowledge, such as the use of image artefacts to represent the various agents and agencies within the system.

# 5 LSD modelling

In order to develop a more concrete idea of the underlying system structure it is useful to develop a high level model of each agent with definitions for the agents interactions with the world. These include external state observables, (what the agent knows about the world), what it can directly affect (what it can change) and what it can perceive (sense about its environment). In addition to this LSD specifications encourage the definition of deterministic internal state protocols, such as 'have task & resource free -> assign task' (Beynon, 2004).

The LSD specifications for this model can be found in the model documentation.

# 6 Model Implementation

In order to fully explore the state space of an empirically created model it is often helpful to implement it using the set of standardised tools, 'Eden'. These tools allow the model, as defined through LSD specifications to be fully tested in order to provide the models observer with an interactive environment for exploratory learning.

In order to create a viable and easy to use interface for the model it is necessary to implement a strategy of information hiding and grouping. As such he interface should only show system agents, their external links (oracles and handles) and artefacts which may help the observer connect with the simulation. This can best be achieved through identification of observables within the system, such as the transfer of tasks and length of task queues at each agent.

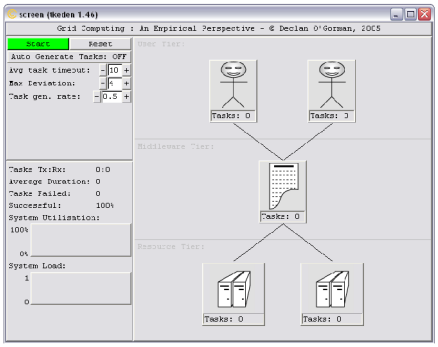


Figure 3 : grid system implemented in tkeden

To ensure an interesting simulation and interaction within the model it is necessary to introduce a random element to help model real world non-determinism, for example through randomising the time intervals at which tasks are dispatched or assigned. Artefacts must also be included to aid the observer in understanding the system processes, examples of some standard artefacts are showing in Figure 4.

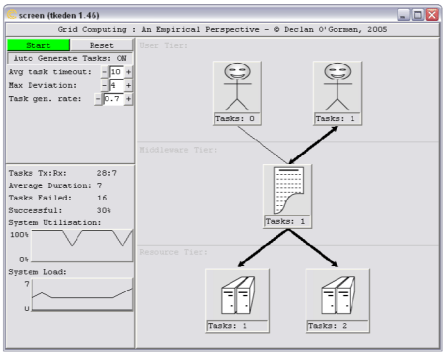


Figure 4: grid computing system in operation

One of the large implementation differences between *tkeden* and various other modelling solutions is the concept of dependency inherent throughout the language. This dependency allows for system wide change from the alteration of a single variable state. The model will be implemented sticking to best practice principles regarding dependencies.

The model attains a simple view of grid with the use of its hierarchical layout and common artefacts in order to convey the underlying state of the system to the observer with the minimum amount of information corruption or misinterpretation.

# 7 Conclusions from model

Through experimentation and interaction with the model it has been possible to draw several conclusions about the original domain of grid computing.

It is necessary to take into account the time taken to return a task to its originator when agreeing to honour quality of service agreements. For large tasks it may take some time for the output data to be returned to the source.

It is necessary to define task processing requirements in some 'normalised' form so that the estimated time to completion can be calculated for various resource centres. This area of research is being covered by the Warwick high performance systems group.

# 8 Evaluation

## 8.1 Evaluation of empirical principles

The empirical principals in use during the modelling of the grid domain have proven to be extremely useful and allowed a generalised overview and model of the system worrying about the low level details. Agentification of the system and use of LSD to specify agent attributes lead to an easy to implement system which retains main of the key artefacts of the real implementation. The abstraction of the solution to a higher level allowed the model to cover a more global view of the system state whilst ensuring the system modeller retained access to selected lower level variables (such as task parameters). Finally the use of animation allowed the system to be tested and observed in a continuous state.

The principals however do place a lot of emphasis on agents, with almost no comment upon the role of objects to these agents. The objects in this case referring to items within the closed world simulation which are incapable of affecting change by themselves, instead they have change inflicted upon them by the system agents. In the domain in question this could be the tasks which are being passed around the system.

Whilst overall the empirical principals provide an interesting and unique view upon a complex domain the use of abstraction and agentification can often eliminate crucial peripheral objects within the system.

## 8.2 Evaluation of the tools

The Eden toolkit, provided for the purpose of modelling derived systems has several large advantages over other similar toolkits. This includes the ability to perform fast prototyping and on the fly redefinition of the system state. The ability to perform this level of in-depth experimentation during runtime is extremely useful for reclassifying the modeller's views and impressions of the system agents and objects.

However the toolkit does have several flaws in comparison to alternative solutions. The lack of any object orientation and modularisation is a hindrance as the use of an object would be highly useful for grouping information (such as tasks) or procedures (such as agents) into one entity within the model.

Eden is therefore primarily bound to a static domain, whereby expansion of the number of non pre-defined agents within the system is extremely difficult and can lead to large levels of complexity and unreliable code relying mainly upon the 'execute()' command which can cause unreliable code, breaking variable dependencies. This is one of the main advantages held by C++, Java and other mainstream modelling solutions.

One of the main uses for the tools is to model non-deterministic agent interactions. As such it seems sensible to include a notion of agent concurrency in the heart of the toolkit, however this is not the case.

The toolkit comprises three languages, each of which uses different standards, adopting different naming conventions (*int* vs. *integer*), line de-limiters (\n vs. *;*) and primitive support. In addition, the use of lists as dependant variables is not very well supported within the tools, whilst the new *%edensl* language does attempt to remedy this situation its implementation isn't well documented and it is unclear what happens to the variables created to reference array elements once the array (list) has been deleted.

Through this model it can be seen that the tools do create a good platform for basic modelling, however in order to mature it will be necessary for the tools to expand in the areas of object orientation, domain expansion and automatic agent detection / exploration.

# 9 Future Directions

There are several directions open for the expansion of this grid model in the future. These include:

- Dynamic agent addition and removal during run-time
- N-brokers including lateral task assignment between brokers

Dynamic state optimisation to determine to optimal attributes for a resource centre given the current environment

Concurrent evaluation and human interaction through the use of the dtkeden tools.

## 10 Conclusion

In conclusion it can be seen that empirical modelling techniques are highly effective with the grid paradigm due to its concurrent, multi-agent structure. It can be seen through the project that the authors bias has made an effect upon the end result and in order to fully complete the conceptual model it would be necessary to consult with users about their viewpoint on the system.

## Acknowledgements

Meurig Beynon, for his invaluable advice and feedback about the project focus and direction.

## References

WM Baynon, SB Russ, *Empirical Modelling of Requirements* (VCCS), University of Warwick, 1995

WM Beynon, *The LSD notation for communicating systems*, CS 55#87, University of Warwick, 1986

*Globus Project: Open Grid Services Architecture,* http://www.globus.org/ogsa/, [Accessed: 05/01/05]

*(further references can be found in the model documentation)*