Distributed Gridflow Model and Implementation

Cheng Bo, Qihe Liu, and Guowei Yang

College of computer science and engineering,
University of Electronic Science and Technology of China,
Chengdu 610059, P.R. China

Abstract. In this paper, we proposed distributed and adaptive grid workflow net model. Which applies the Coloured Petri net as the formalism to describe grid process, and proposed the formal method for grid services to composite the gridflows. And also proposed the multi-agent based implementation for gridflow net model.

1 Introduction

The Open Grid Service Architecture (OGSA) tries to address the challenge to integrate services spread across distributed, heterogeneous, dynamic virtual organizations [1]. Which required to composite grid services into gridflows to gain grid goals. Gridflow faces many uncertain factors such as unavailability, incomplete information and local policy changes [2]. Therefore, we proposed the coloured petri net based adaptive gridflow net model.

2 CPN Based Gridflow Net Model

CPN [3][4] which extends the formalism of classical Petri net [5] by allowing a token to be of a specific distinguished colour. The sound mathematical foundation behind the CPN makes it a very useful tool for model distributed systems.

Definition 1. (Coloured Petri Net, CPN) A CPN is a tuple,

$$CPN = (\Sigma, P, T, A, N, C, G, E, I) \tag{1}$$

(1) Σ is a finite colour set specifying the type of tokens.
(2) (P, T, A) means basic Petri Net. P is the places set, T is the transitions set, $A \subseteq T \times P \cup P \times T$ is the acres set.
(3) N is a node function defined from A into $P \times T \cup T \times P$.
(4) C is a colour function defined from P into C.
(5) G is a guard function from T to expression as such,
$$\forall t \in T{:}[Tpye(G(t)) = Boolean \wedge Type(Var(G(t))) \subseteq C]$$
(6) E is an arc function from A to expression as such,
$$\forall a \in A{:}[Tpye(E(a)) = C(p(a)) \wedge Type(Var(G(a))) \subseteq C]$$
(7) I is an initialization function from A into expression as,
$$\forall p \in P : [type(I(p)) = C(p)_{MS}]$$
where,$C(p)_{MS}$ denotes the multi-set over a set,$Type(v)$ denotes the type of a variable , $var(exp)$ denote the type of variable in expression.

H. Jin, D. Reed, and W. Jiang (Eds.): NPC 2005, LNCS 3779, pp. 84–87, 2005.

Definition 2. (Gridflow Net, GFN) A Distributed Gridflow Net is a tuple,

$$GFN = (CPN, i, o) \tag{2}$$

(1) IN and OUT are subsets of P, IN which has one element is a set of workflow start places, and OUT which may have one or many elements is a set of terminal places formal description. $IN, OUT \in P: \mid IN \mid = 1, \mid OUT \mid \geq 1$, and $\forall i \in IN, {}^\bullet i = \Phi; \forall o \in OUT, o^\bullet = \Phi$.
(2) $\forall x \in P \cup T \wedge x \in IN \wedge x \in OUT, x$ is on the path from $i \in IN$ to $o \in OUT$.

3 The Algebra of Gridflow Composition

Basic gridflow structure as sequence, parallelism, choice and iteration are foundations to composite the gridflow. The grammar is as such,

$$Grammar ::= GS \propto GS | GS \circ GS | GS \parallel GS | \lambda GS \tag{3}$$

where, GS represents a atomic grid service. $GS \propto GS$ represents sequence structrue which is defined as an ordered series of grid services. $GS \circ GS$ represents a choice structure which is selected to execute grid services at run-time when its associated conditions are true. $GS \parallel GS$ represents a parallel structure which means the grid services are performed concurrently. λGS represents a iteration structure which means to perform a grid service for certain number of times. The basic gridflow structure can be used to construct many complex grid workflows.

4 Formal Composition for Grid Services

Grid service can be mapped into a Petri net [6]. The basic control flows of grid services composition is represented as sequential,parallel, choice and repetition.

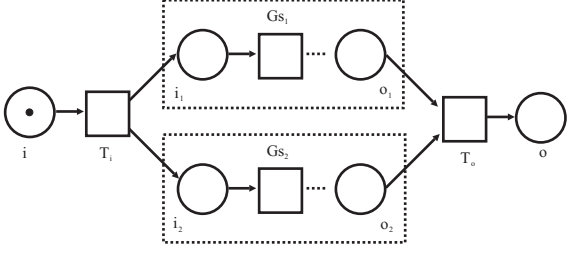

Fig. 1. The parallel composition for grid services

The parallel grid services composition $GS \parallel GS$ is as in Figure 1. Defined as,

$$GS_1 \parallel GS_2 = (ID, GSD, GSP, GRS, GFN) \tag{4}$$

where ID is the parallel grid workflow symbol. GSD is the grid services domain. GSP is the grid services prividers. $GRS = GR_1 \cup GR_2$ is the resources set which required by parallel grid workflow. $GFN = (CPN, i, o)$, where $\Sigma = S_1 \cup S_2$, $P = P_1 \cup P_2 \cup \{i, o\}$,$N = N_1 \cup N_2$, $T = T_1 \cup T_2 \cup \{T_i, T_o\}$, $A = A_1 \cup A_2 \cup \{(i, T_i), (T_i, i_1), (T_i, i_2), (o_1, T_o), (o_2, T_o), (T_o, O)\}$

5 Gridflow Net Model Implementation

Multi-agent based gridflow implementation built on Globus Toolkit and Aglet platforms. Globus Toolkit based on open-standard grid services which provide for resource monitoring, discovery, and management, security and so on. Aglets is a Java environment for mobile agents [7] development and implementation which is designed to exploit the strengths of platform independence, secure execution, dynamic class loading, multithreaded programming and object serialization[8]. The gridflow net implementation is as Figure 2.

Gca represents gridflow composition agent. Which is responsible to composite formal grid services into gridflow based on colour petri net, and also contains all necessary related information to be required by the grid workflow control agent. Mma represents monitoring mobile agent. Which is responsible for monitoring the grid nodes fault status periodically. The planning and checking algorithm [9] is embeded into Mma to detect the grid nodes fault status and share the status among grid domains. Wca represents gridflow control agent. Which is responsible for scheduling and rescheduling grid services to re-composite the gridflow dynamically. When demands from predefined gridflow occur, monitoring mobile agent can detect the deviations, then migrate to workflow control agent server to negotiate and decided which grid services should be invoked with the DAML-

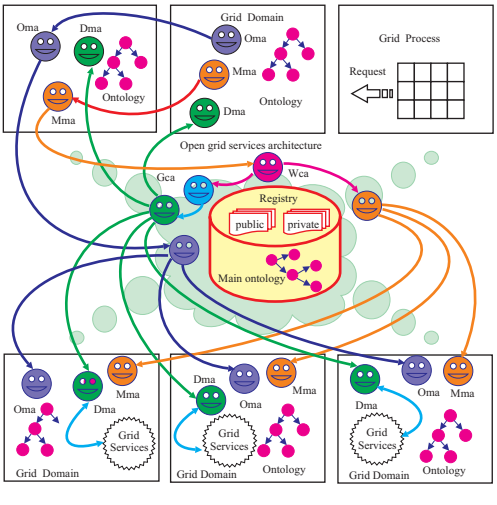

Fig. 2. Gridflow Net Model Implementation

S based ontology to adapt the dynamic changing grid. Then pass the related information to dispatch mobile agent. *Dma*, represents dispatch mobile agent. Which is responsible for interpreting the gridflow definition and controls the instantiation of gridflow processes. Which can migrate from one grid domain to the other domain to execute the corresponding grid services specified by *Gca*. *Oma*, represents ontology mobile agent. Different grid domain has different ontology. *Oma* can translate the similar concepts among grid domains. *Oma* can move to different grid domains to exchange and share the domains knowledge to make the simlar concepts understandable in different grid domains.

6 Conclusions and Future Work

We mainly introduce the Coloured petri net based distributed Gridflow process model and describe the formal composition for Grid services. Also proposed agent-based adaptive Gridflow implementation. Gridflow security is a another key problem. In the future, we will focus on the access control for the gridflow.

References

1. Foster, I, et al: "The Physiology of the Grid: An Open Grid Services Architecture for Distributed Systems Integration". OGSA WG Global Grid Forum, June (2002)
2. Jia Yu and Rajkumar Buyya: "A Novel Architecture for realizing GridWorkflow using Tuple Spaces". In proceedings of the Fifth IEEE/ACM International Workshop on Grid Computing. (2000) 119-128
3. K. Jensen: "Coloured Petri Nets. Basic Concepts, Analysis Methods, and Practical Use". Monographs in Theoretical Computer Science. Springer Verlag. 1-3(1992-1997)
4. Dongsheng Liu: "Modeling workflow processes with colored Petri nets". computers in industry. 49(2002) 267-281
5. Zhijie Guan, Francisco Hernandez, and Purushotham Bangalore: "Grid-Flow: A Grid-Enabled Scientific Workflow System with a Petri Net-Based Interface". Accepted for publication in Concurrency and Computation: Practice and Experience, Special Issue on Grid Workflow. 2005
6. Rachid Hamadi, Boualem Benatallah: "A Petri Net-based model for Web service composition". In Proceedings of the 14th Australasian Database Conference. Australian Computer Society. February (2003) 191-200
7. Juan R. Velasco1 and Sergio F. Castillo: "Mobile agents for Web service composition". In Proceedings of 4th International Conference on E-Commerce and Web Technologies. Lecture Notes in Computer Science. september (2003) 135-144
8. Ian Gorton, Jereme Haack, and David McGee,et al: "Evaluating Agent Architectures: Cougaar, Aglets and AAA". Lecture Notes in Computer Science. Springer-Verlag. 2490(2004) 264-278
9. Alexander Lazovik: "Planning and Monitoring the Execution of Web Service Requests". Proceedings of international conference of Service-Oriented Computing. June (2003) 335-350