Adding domain-specific constructs to Event B for developing and reasoning about grid applications

Pontus Boström and Marina Waldén
Åbo Akademi University

Grids

- Used for large-scale distributed systems
  - Scientific computing, e.g., in Physics and engineering
  - Business applications
- Share information and computational resources over organizational boundaries
- Typical grid application needs:
  - Virtual Organisation management (who participates, resources contributed, resources used, etc.)
  - Resource discovery and management
  - Job management
  - Security and data management to support all the services

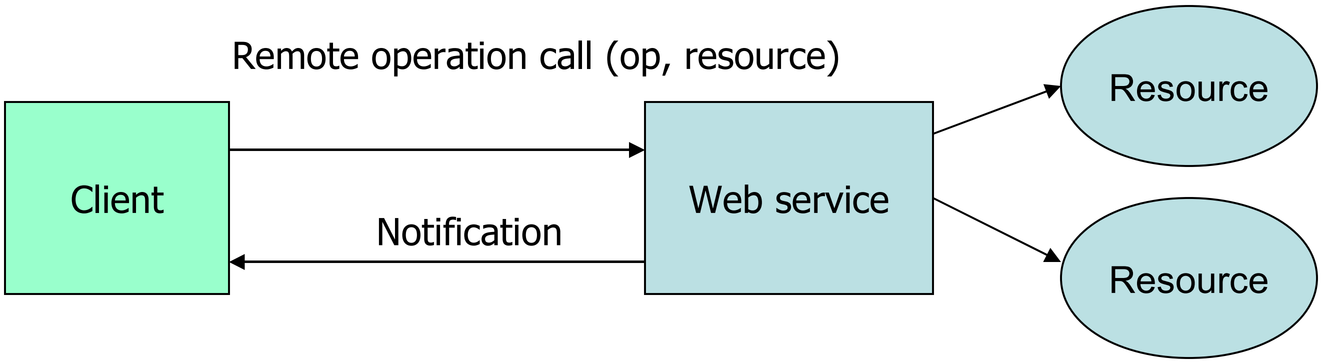
Open Grid Services Architecture (OGSA)

- Defines the basic services required for grid enabled applications
- Service-oriented
  - Everything implemented as services with standardised interfaces
- Based on Web services
  - OGSA requires stateful services
  - Web services traditionally stateless
- Web Service Resource Framework (WSRF)
  - Standard for stateful web services
  - Standardised by OASIS
  - Services similar to remote objects in CORBA and RMI

Åbo Akademi University

Web Service Resource Framework

- Based on Web Services
  - XML
  - SOAP
  - WSDL
- Extends Web services with
  - State (WS-Resource)
  - Potentially transient services (WS-ResourceLifeTime)
  - Notifications (WS-Notification)



The Globus Toolkit

- Toolkit for developing grid applications
  - Implements many of the OGSA services
  - De-facto standard
- Implements and uses WSRF
  - Stateful web services
- Most services available as WSRF services
  - Job management
  - Resource management and discovery
    - Managing information in the grid, e.g., available services
  - Secure file transfer
- Security infrastructure also available

Grid implementations

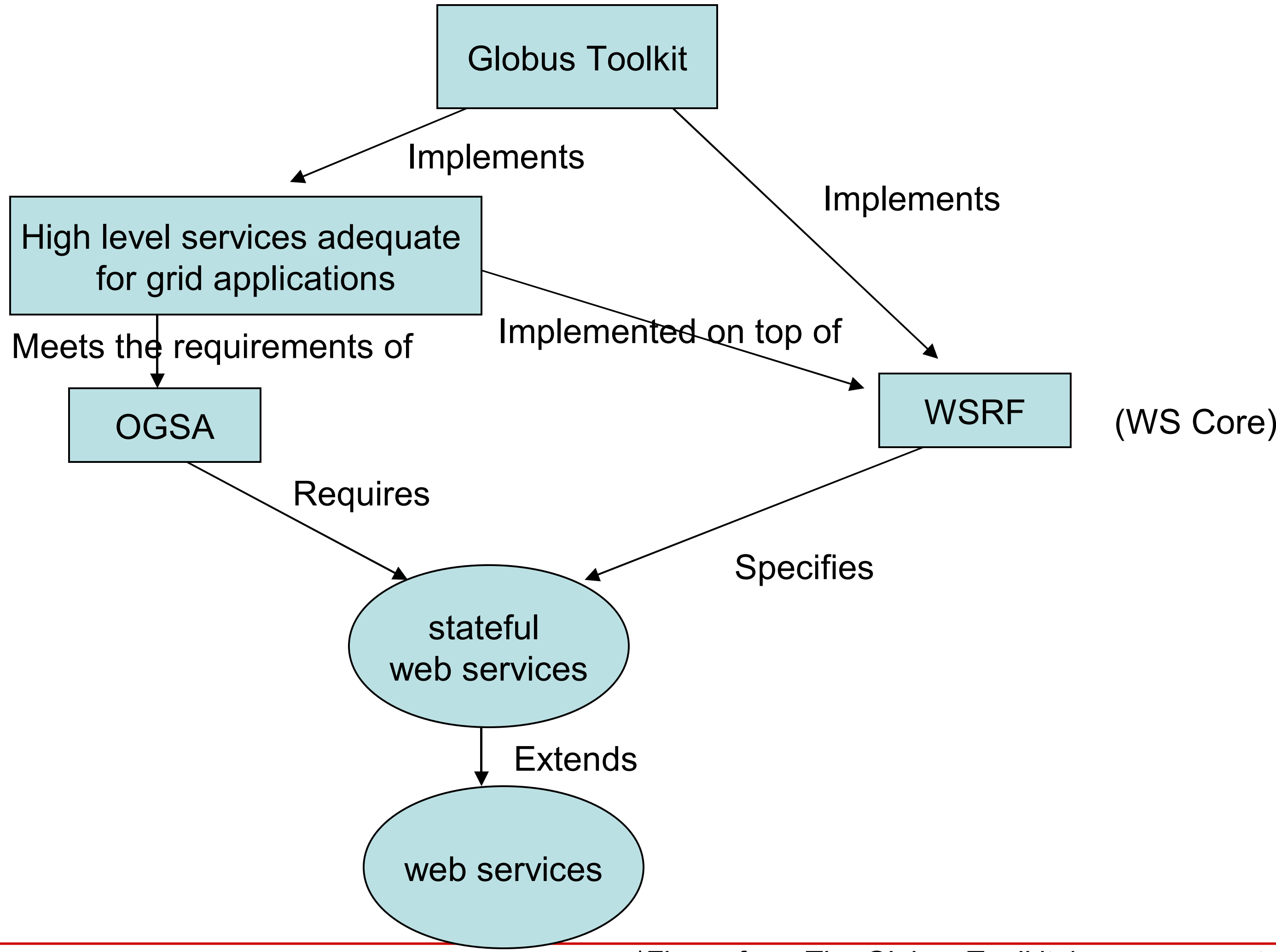


*Figure from The Globus Toolkit 4 programmer tutorial: http://gdp.globus.org/gt4-tutorial

Need for formal methods

- Difficult to implement "correct" Grid applications
- Formal methods useful in order to develop correct specifications
  - Can be difficult to implement
- The specification language should take into account the features of the underlying platform
  - Specifications easier to understand, since they can clearly talk about domain-specific concepts
  - Specifications are potentially easier to implement

Event B

```
SYSTEM C
SEES
  C_CTX
VARIABLES
  x
INVARIANT
  I(c,x)

EVENTS

INITIALISATION =
      Si(c,x)
Evt1 =
      ANY u WHERE G1(c,u,x)
      THEN S1(c,u,x)
      END;
Evt2 =
      WHEN G2(c,x)
      THEN S2(c,x)
      END;
END
```

- Modification of the B Method for development of reactive, distributed or concurrent systems
- Developed by J. R. Abrial
- Based on Action Systems by Back and Kurki-Suonio

- Centered around the notion of refinement
  - Start from a initial specification that takes into account the most important requirements
  - Develop it stepwise through refinement steps towards a more concrete and implementable model

Formal development of Grid applications

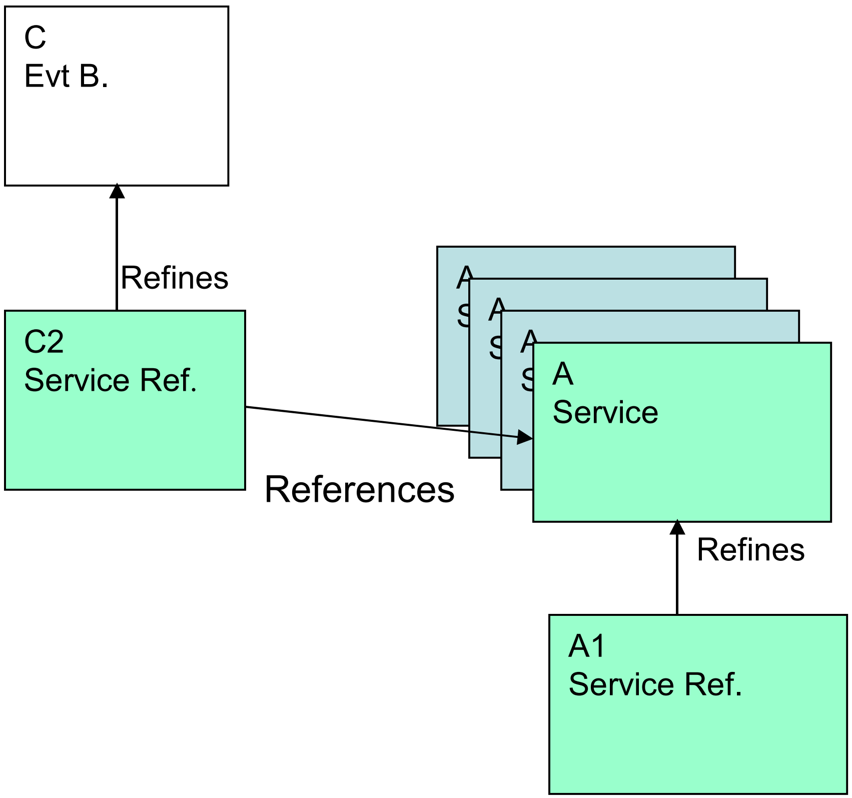
- We like to have a formal method suitable for developing grid applications
- Difficult to create implementable specifications of grid applications in Event B
  - No grid communication mechanisms such as remote operations and notifications
  - Difficult to implement due to synchronization issues and the atomicity requirement of events
- We have extended Event B with constructs for
  - Specifying stateful (grid) services
  - Remote operation calls and notifications
- Extensions should be introduced in a manner that simplifies implementation

Åbo Akademi University

Grid extensions to Event B

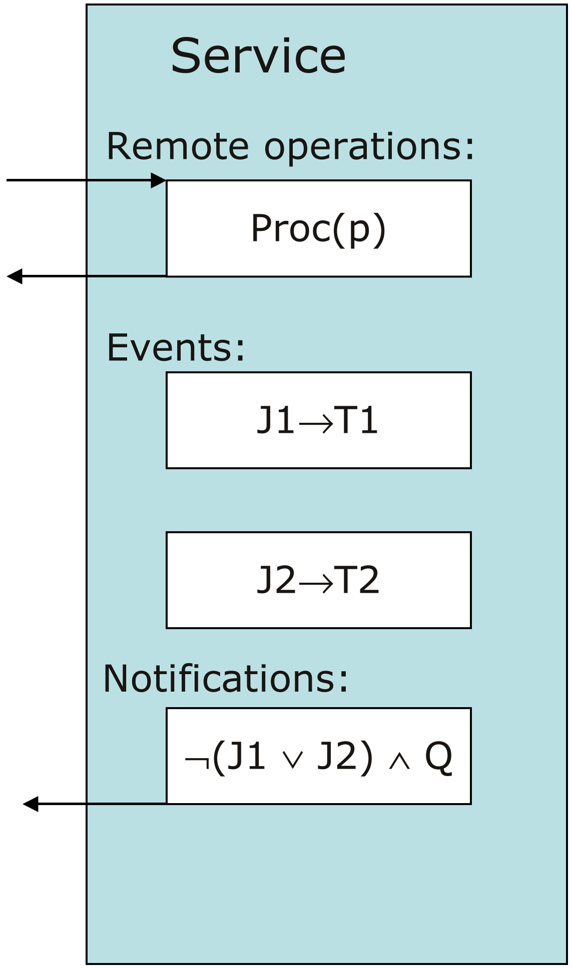
- Provides two new types of B machines
  - SERVICE
  - SERVICE_REFINEMENT
- Take into account grid specific features
  - services with state
  - Remote operations
  - Notifications
- Enables proofs of properties about the entire system
- Are translated to ordinary (Event) B for verification
  - Automatic generation of proof obligations

Development overview



Grid service machine

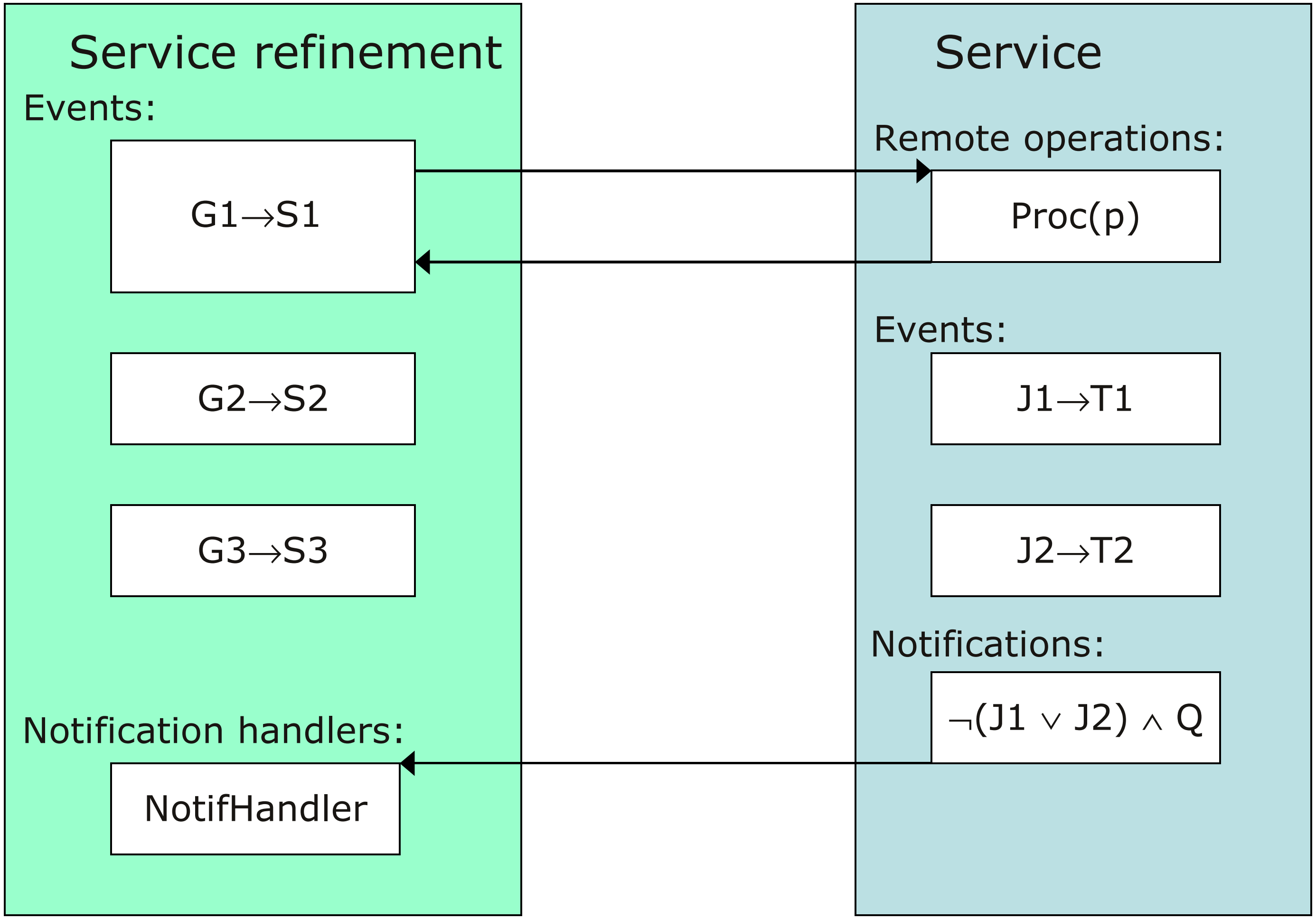
- Abstract specification of a (WSRF) service
- A service machine is a template that clients obtain instances of
  - Compare to Classes in OO
- Remote operations
  - Ordinary B operations called from a client
- Events
  - Executed independently of a client
- Notifications
  - Sent when all events have become disabled



Grid refinement machine

- A client that calls remote operations in grid service machine instances
- Refines
  - Event B machine
  - Service machine
- Clause for enabling dynamic management of grid service machine instances
  - Instances are used as variables (cf. OO in B e.g. UML-B)
- Clause for refining remote procedures
- Clause for refining events
- Events for handling notifications
  - New events
  - Enabled when a notification has been sent from a service
  - Executed once every time a notification is received

The behaviour of grid components



Service machine

```
SERVICE A

VARIABLES
  y
INVARIANT
  Inv_A
INITIALISATION
  y := y0

REMOTE_OPERATIONS

   Proc(p) =
     PRE P(p)
     THEN T
     END
```

```
EVENTS

   A_Evt1 =
     ANY u WHERE J1(u,y)
     THEN T1
     END;

   A_Evt2 =
     WHEN J2
     THEN T2
     END;

NOTIFICATIONS
   Notif =
     GUARANTEES Q END

END
```

Service refinement machine

```
GRID_REFINEMENT C2
REFINES C1

REFERENCES A
VARIABLES
  z,x,a_inst

INVARIANT
  a_inst:A &
  Inv_C

INITIALISATION
  x := x0 || z:=z0 ||
  a_inst::A
```

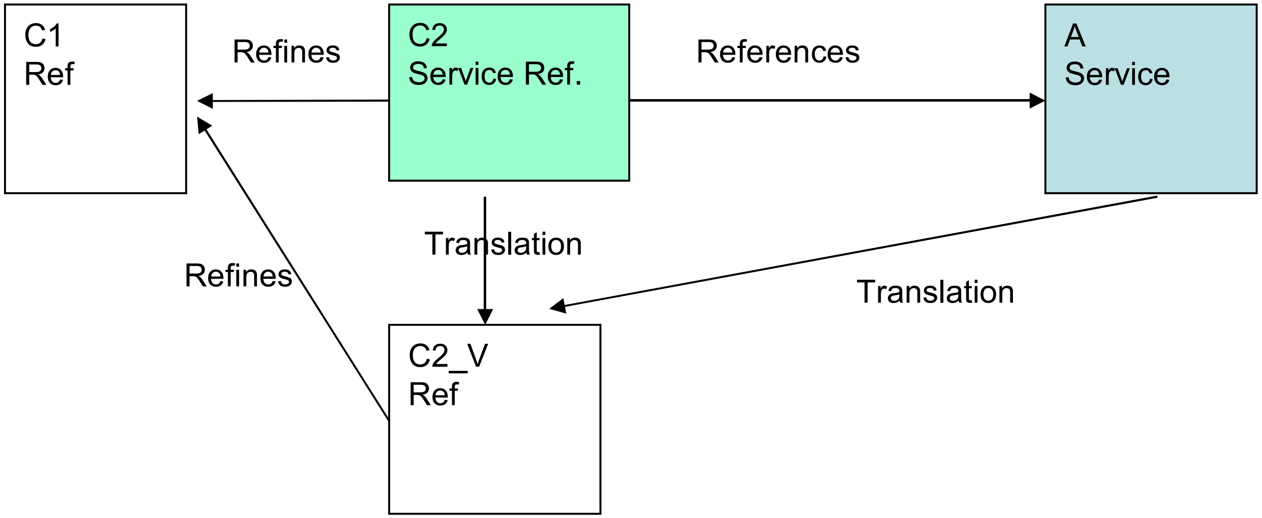
```
EVENTS
  C_Evt1 =
      WHEN G1'
      THEN a_inst.Proc(x) || S1'
      END;
  C_Evt2 =
      ANY u WHERE G2'(u,x)
      THEN S2'
      END;

NOTIFICATION_HANDLERS
  NotifHandler =
      NOTIFICATION Notif
      SOURCE v:A
      THEN S3
      END
END
```

Proof obligations

- The following proof obligations need to be generated in order to show that an event system is a refinement of another:
  - Refinement of the initialisation
  - Refinement of each event
  - Introduction of new events
  - Termination of new events
  - Deadlock freeness

Verification overview

- The semantics of the new constructs are given by their translation to (Event) B
  - Enables reuse of existing proof tools



Verification

- The events of the SERVICE machine are merged with the events in the client
- The variables of the SERVICE machine are translated to functions from instance to variable type
- The remote operations calls are inlined
- A notification is sent only after all events in the grid service have become disabled
- The notification handler is only be executed once for each notification

Code generation and Tool support

- The domain-specific constructs of Event B ensure that the specification can be implemented in a grid environment
- Grid enabled code can be generated from the specifications
  - Code for the Distributed B specification
  - Code for setting up connections in the grid environment
  - Interface description of the services in WSDL
- No tool support
  - Tool needed for translating to B
  - Tool needed for translating to Java

Conclusions

- Enables construction of correct grid applications
- Automatic generation of proof obligations
- Implementable architecture by construction
  - Although not very flexible
- These Event B extensions can also use other middleware for distributed systems

Åbo Akademi University

References

- Pontus Boström and Marina Waldén. An Extension of Event B for Developing Grid Systems. In *Proceedings of the 4th International Conference of B and Z users - ZB2005: Formal specification and Development in Z and B*, Apr 2005.

- Pontus Boström and Marina Waldén. Development of Fault Tolernt Grid Applications Using Distributed B. In *5th International Conference on Integrated Formal Methods, IFM2005*, Dec 2005.