Virtual Simulation Objects Concept as a Framework for System-Level Simulation

Sergey V. Kovalchuk, Pavel A. Smirnov, Sergey S. Kosukhin, Alexander V. Boukhanovsky
e-Science Research Institute, National Research University ITMO, St.Petersburg, Russia

IEEE International Conference on eScience - 2012
Chicago, USA
Agenda

- Complex simulation tasks
- Virtual Simulation Objects concept
- Workflow composition and running
- Domain knowledge sharing
- User interfaces
- Future works and conclusion
Complex Simulation Tasks
Modern eScience tasks

- Great diversity of the resources
  - Different types of resources to be integrated: software, hardware, data, human resources
  - Different low-level access and usage procedures for resources of the same type
- Domain specialists as end-users
  - Domain-specific terms as a language
  - Low level of technological background
- Simulation task complexity
  - Complexity of simulated system
  - Uncertain requirements of simulation

High-level user support required during description of complex system simulation process
Composite application

**Ideal world**

- Users are domain specialists

- There are well annotated domain-specific services and parameters

- There are tools for service composition and running
Composite application

Complicated world

- Users are domain specialists
  - Multidisciplinary tasks - many domains
  - Users with weak domain background
- There are well annotated domain-specific services and parameters
  - Set of IT-related parameters
  - Complex domain-specific parameters with dependencies and variations
  - Complex workflows to solve the tasks
- There are tools for service composition and running
  - Tools are mostly “IT-centric”
  - Low-level access to the services
  - System-level science as a requirement
Domain-specific way

- Composite solution constructed using domain concepts
  - Domain knowledge usage for task description
  - High-level user support within composition process
  - Domain specific terms translation

Especial importance for multidisciplinary tasks
Virtual Simulation Objects
Basic requirements

VSOs are building block which allow

- **Construct a system description** (structural model)
- **Express and distribute domain-level knowledge** on object’s simulation
- **Run simulation** process upon constructed structural model
- **Join modeling, simulation and results analysis** processes
- **Visual representation of simulated system**
iPSE approach

iPSE - Intelligent Problem-Solving Environment

Software environment for strong user guidance during whole process of problem solving

- Decision support for selecting solution methods
- Intensive use of knowledge
- Using target problem domain language
- Rich user interface
- Hiding the software and hardware details
- Utilization of heterogeneous soft- and hardware
VSO visual structure

Baltic sea: Sea

Object processing mode
- forecast
- static analysis
- parameter optimization

Bases pane
define available bases for parameters (space, time, group)

Object parameters are available for every model in the object

Data status
- correct (OK)
- Need to be (re-)defined (?)
- unavailable (X)

Dataset's basis
from bases pane (one from each group)

Data sets
- can be obtained from models internal or external for object
- additional setting mode: storage, external services, manual setting
- are checked for correctness and consistency

Models
- are transformed into parts of composite workflow for running simulation
- can be tuned: Software and scenario selection, parameter setting
- can be switched off by the user

Object parameters
- are available for every model in the object

FG/Ø

Bathymetry

Source: storage

Aloft wind

Source: BOOS

Sea waves

Scenario: Shallow water
Software: Swan

Near-water wind

Scenario: From aloft wind
Software: NWW_ITMO

Wave spectrum

Source: model

FG/FT

Workflow... Parameters...

Bases pane
define available bases for parameters (space, time, group)

Object parameters are available for every model in the object

Data status
- correct (OK)
- Need to be (re-)defined (?)
- unavailable (X)

Dataset's basis
from bases pane (one from each group)

Data sets
- can be obtained from models internal or external for object
- additional setting mode: storage, external services, manual setting
- are checked for correctness and consistency

Models
- are transformed into parts of composite workflow for running simulation
- can be tuned: Software and scenario selection, parameter setting
- can be switched off by the user
Workflow Composition and Running
CLAVIRE – basic platform

- **CLAVIRE - cloud application virtual environment**
- Platform for abstract composite application execution
- Different computational infrastructures are integrated
- Domain specific language (EasyFlow) for composite application description
- Use expert knowledge for:
  - Workflow composition
  - Task processing
  - Data analysis

---

**System structure**

- Intellectual subsystem
  - Task processing
  - Service monitoring
  - Data visualization subsystem

- Task and workflow storage
  - Running commands
  - Alerts

- File storage
  - File characteristics
  - Resource characteristics
  - Running commands

- Metcomputing interface
- Grid interface

---

**Infrastructure**

- Cluster 1
- Cluster M
- Grid infrastructure

---

**Open services**

- Command router
- Web-interface
- Authorisation
- Monitoring
- File transfer
- Visualization streaming

---

**Training materials**

---

IEEE eScience Conference - 2012
Cloud Computing Maturity Model

CCMM

- Consolidation
- Abstraction
- Automation
- Utility
- Market

CLAVIRE

- Resources and infrastructures integration
- Abstract description of composite software with DSL
- Automatic translation and running of composite applications
- Support of workflow composition with domain knowledge
- Place for sharing of software, resources, workflows, knowledge
Workflow approaches

- **Level 1:** Composition of concrete services
- **Level 2:** Composition of abstract services
- **Level 3:** Templates and AWF support tools

**Level 4:** Meta workflow – templates composition

---

**Common approach**

- UI dialog
- Task definition
- Methods selection
- AWF(s) composition
- Resource selection
- Scheduling

---

IEEE eScience Conference - 2012
VSO management system

- VSO management system: class editor, constructor, knowledge base
- Building and running simulation workflows using interactive mode within CLAVIRE
- Integration of data-source services within CLAVIRE
- Connection to virtual reality and upper-level ontologies
Domain Knowledge Sharing
Knowledge within iPSE

- Knowledge groups
  - Domain-specific
  - IT-related
  - Solution design

- Knowledge processing
  - Ontology processing
  - Solution composition
  - Characteristics estimation for solutions

- Performance modeling
  - Models as a knowledge
  - Workflow performance estimation
Knowledge processing

- Four conceptual layers: problem, method, package, service
- Two sub-layers: non-parameterized, parameterized
- Background data analysis
- Dynamic structure composition
- Quality estimation according to set of criteria
Semantic description for VSO

- Ontological description of simulated objects
- Basic semantic structure of simulation process
- Relationship with high-level ontologies for automatic composition
- Ontologies as a subject to share within the library

Library of VSOs semantic description by domain experts
User Interfaces
VSO constructor

- Silverlight web application
- Use OWL ontology structure to generate interface primitives
- Palette of available objects for system description
- Automatically connect values by semantic description
- Allows to export composite application to the CLAVIRE
VSO class editor

- Silverlight web application
- Visually construct VSO classes
- Visual primitives similar to VSO constructor
- Interconnection with semantic bases from CLAVIRE (resources, parameters etc.)
- Export into VSO knowledge base in a form of ontology
Running example

- **Two objects:**
  - Sea: simulation of waves, driven by wind
  - Ship: simulation of behavior on the waves

- **Simulation run on CLAVIRE**
  - Workflow available within CLAVIRE environment
  - Running software available within the environment

- **Visualization within 3d-scene**
  - Basic visual scene: the ship on the sea
  - Simulated data: forces driving the ship
Future Work and Conclusion
Feature works

- Deeper automation with virtual simulation objects:
  - bases mapping
  - connecting to high-level ontologies (SUMO, CYC, etc.)
- System-level task solving using VSO structure
  - object’s parameters optimization
  - system’s structure optimization
  - decision support using expressed knowledge
- Integration with virtual reality
  - general solution for building 3d-scenes
  - objects’ composition within 3d-scene
  - knowledge acquisition with brain-computer interface for
Conclusion

- Virtual Simulation Objects (VSO) approach allow to solve complex simulation tasks by building and running composite applications in more obvious and easier way.
- It become possible to share knowledge on simulation using VSO elements within knowledge library which is built as ontological structure.
- Abstract graphical interfaces allows to work with the system without technological background by manipulation with simulation graphical primitives.

Hiding all the technological knowledge behind the conceptual interactive interface allows to shift further from “IT-specific” point of view to the domain-specific simulation point of view.
Thank you!