Grant Agreement No.: 604656

Project acronym: NanoSim

Project title: A Multiscale Simulation-Based Design Platform for Cost-Effective CO_2 Capture Processes using Nano-Structured Materials (NanoSim)

Funding scheme: Collaborative Project

Thematic Priority: NMP

THEME: [NMP.2013.1.4-1] Development of an integrated multi-scale modelling environment for nanomaterials and systems by design

Starting date of project: 1st of January, 2014

Duration: 48 months

WP N°	Del. N°	Title	Contributors	Version	Lead beneficiary	Nature	Dissemin. level	Delivery date from Annex I	Actual delivery date dd/mm/yyyy
1	D1.2	Detailed Workflow Document	Author: Thomas F. Hagelien Checked by: Stefan Radl (TUG) and Christoph Kloss (DCS)	0	SINTEF	Other	CO	31/06/2014	18/08/2014

Comment: This report is a first version of work in progress and will be updated by the end of the year in 12M, 24M, 36M and 48M.

1 Introduction

NanoSim can be considered an ecosystem of simulators and tools that collectively form a multi-scale simulation tool that requires that data propagates and transforms through a pipeline of automatic and manual operations from the lowest to the highest scales.

1.1 Purpose

The purpose of this report is to document the workflows and information sharing between different simulators in NanoSim, to ensure that we have all the information we need in the exchange of data, and that the data is compatible with the different software that requires it. The workflows described in this document are based on the Use Cases defined in *Functional Requirements Documentation [WP1, D1.1] Chapter 3.* Some of these Use Cases only address the isolated operations in a single simulator, while other involves more than one simulation tool to achieve a goal.

1.2 Intended readers

The indented readers of this document are developers and scientists that are working with the offline coupling of models, who need to understand which models are being connected, and what data is shared.

1.3 Overview

This document is divided in two main parts. Chapter 2 is dedicated to specification of the data (meta-data) that will be part of the NanoSim delivery D1.4 *(Meta-data-files written as formal schemas in JSON)*. The second part dives into the workflows – where the high-level workflow (connectivity) of NanoSim is described along with more detailed UML (Unified Modeling Language) activity diagrams.

1.4 Assumptions

Workflows are traditionally described using Workflow Diagrams, while data/information flows are described with Data Flow Diagrams. In UML so-called Activity Diagrams are used to model computational processes (workflows), while the *data flow* can be modelled with Interaction Diagrams. We are, however, not interested in the details of the information exchange itself, but rather in the identification of the properties of the data that are to be shared/exchanged. Therefore, we will use Activity Diagrams to illustrate the dynamic aspects of the Use Cases, and explicitly describe the data in terms of entities with its associated properties.

It is assumed that the basic concept of Entity-Relationship modelling is understood, as Porto uses this data model in the implementation of the data-centric architecture.

All data that will be exchanged should be defined in terms of properties of entities, and collections of entities are specified in terms of relationships between the different entities.

1.5 Issues

As the NanoSim project is still in an early phase, it is too early to define the specific information of the metadata. The 6 month consortium meeting and workshop in London June 2014 the consortium started the work to map out all data/information transferred between the different simulators in the NanoSim ecosystem.

It is the intension to continuously iterate and improve this document, to accurately reflect the available level of detail of the transferred data/information.

2 Meta-data

In order to verify the completeness of the data being shared by multiple applications we need to agree on a minimum level of information about the data (meta-data). All entities should eventually be defined with the template illustrated in Table 1. Alternatively, entities should be defined using a formal specification language as detailed in WP1.

Table 1 Entity specification template

Entity Identification									
Name									
Version									
Description									
Dimensions									
Name		**Description**							
Name		**Description**							
Enumerators									
Name		**Value**		**Desc.**					
Properties									
Name		**Type**		**Unit**		**Dim**		**Desc.**	
Name		**Type**		**Unit**		**Dim**		**Desc.**	
Name		**Type**		**Unit**		**Dim**		**Desc.**	
Name		**Type**		**Unit**		**Dim**		**Desc.**	

2.1 Meta-data fields

2.1.1 Entity Identification

This should uniquely identify the entity such that any application that refers to a given entity name with a given version number can be assured that this description will never change. Any modification to the entity requires either a new entity name, or a revised version number of that entity.

2.1.2 Dimensions

This field defines a physical dimension (for instance '*number of grid cells in I-direction*') defined by a name and a length (a positive integer). Dimensions need to be used to define properties that are represented by arrays. For example, the velocity of *N* bubbles in *nSpatial* spatial directions needs to be represented by an array with dimensions [*N* x *nSpatial*].

2.1.3 Enumerators

This field defines a 'local' enumerated type. An example could be a list of available model-types, while a property named '*model*' can be of the enumerated type. This will enable a user to choose a model from a list of different available options.

2.1.4 Properties

To ensure there are no confusion in the interpretation of a data point, properties should be specified with name, data type, unit, dimensions and a description of the property. The type should be very specific (e.g., the intrinsic data type used in the computer code implementation, e.g., the C-language data type). Units should be defined such as to avoid errors due to ambiguity. Dimensions should be defined using the names specified in the dimensions-field of the entity described earlier. If an array has rank higher than one, a list of dimension names should be used. Column-major order (native Fortran array layout) vs row-major order (native C layout) should in this case be clearly specified.

2.2 Entity Example

To illustrate how an entity can be specified in terms of meta-data, we can use a bubble-column as an example:

Table 2 A bubble column entity

Identification									
Name	BubbleColumn								
Version	0.1-SNAPSHOT-1								
Description	Bubble column properties								
Dimensions									
Name	nSpatial	**Description**	Number of spatial directions (typically 3)						
Name	N	**Description**	Number of bubbles						
Properties									
Name	velocity	**Type**	double	**Unit**	cm/s	**Dim**	[N x nSpatial]	**Desc.**	Steady-state velocity
Name	size	**Type**	double	**Unit**	mm	**Dim**	[N]	**Desc.**	Diameter
Name	area	**Type**	double	**Unit**	mm^3	**Dim**	[N]	**Desc.**	Interfacial area

Table 2 defines the properties bubble-velocity, bubble-size and interfacial area for each bubble in a vector of *N* bubbles. The unit and array sizes are well defined. The type *double* is assumed to be a IEEE754 double-precision binary floating-point format: binary64.

2.3 Entity-Relationship

A group of entities is called a collection (in Porto). Entities that have a semantic dependency to other entities in the same collection is said to have a relationship. The type of relationship and relationship type needs to be defined. Relationships have a defined *direction*. By this we can say that A *contains* B, where *contains* is the relation. This does not imply that B contains A. The template shown in Table 3 can be used to define the entity relationships for each collection.

Table 3 Relationship definition template

Relationships Table				
From (Entity)		**To (Entity)**		**Relationship**
Name	Version	Name	Version	Type

3 Workflows

These are the preliminary results from the group work, where the goal was to map out all data/input between the different simulators in the NanoSim ecosystem. As Figure 1illustrates, the output from the experimental activities will be used as input to the atomistic modelling in WP3. The interactions between the quantum mechanical computation and the Kinetic Monte Carlo simulation are not indicated here. The data exchange from WP3 (Atomistic Modelling) and WP4 (PaScal/COSI) is the link between the atomistic scale and the DNS (CFD level-1) scale. The integration of WP4 results into WP2 is part of the *online* coupling simulator COSI, which couples DNS with the reactive particle model PaScal. The link between WP2 (COSI) and WP5 is the connection between Resolved CFD (CFD-level-2) and Filtered CFD (CFD-level-3). WP6 is the equipment simulation scale (Phenom). The figure also illustrates the manual steps needed to run some of these simulations. Data is categorized as "input", "output" or "input *(from literature*)". The latter is intended to indicate that the data will not be available from previous simulation steps, and needs to be feed manually into the system. Note that this illustration is a very high-level representation of the workflow, and the data flow indicates only key data/information that needs to be exchanged.

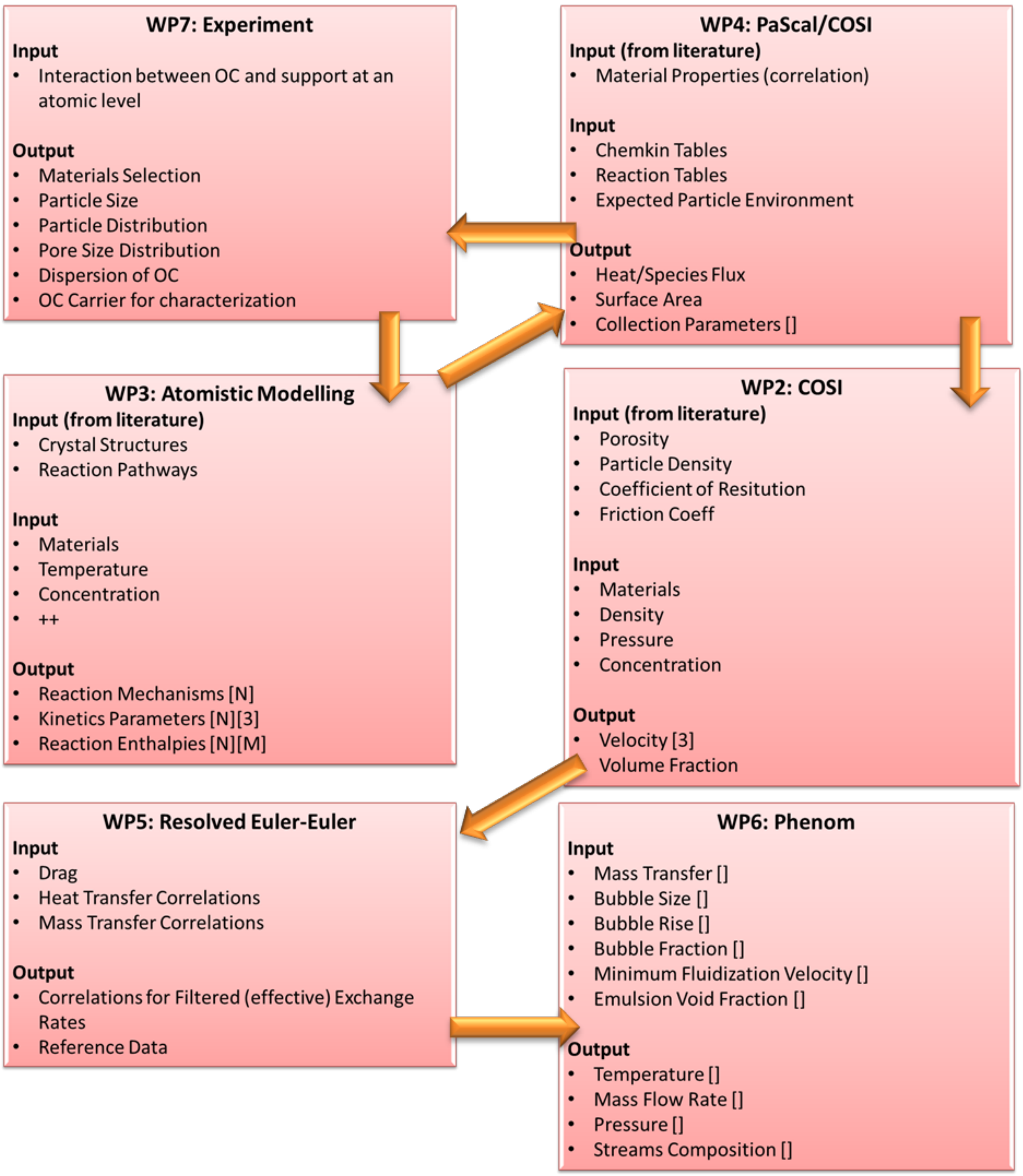

Figure 1 Data information exchange diagram

The following section contains the UML Activity Diagrams for the currently defined Use Cases. [See *Functional Requirements Documentation Ch.3]*

3.1 Generate a Filtered Drag Model incl. Reversibility Check

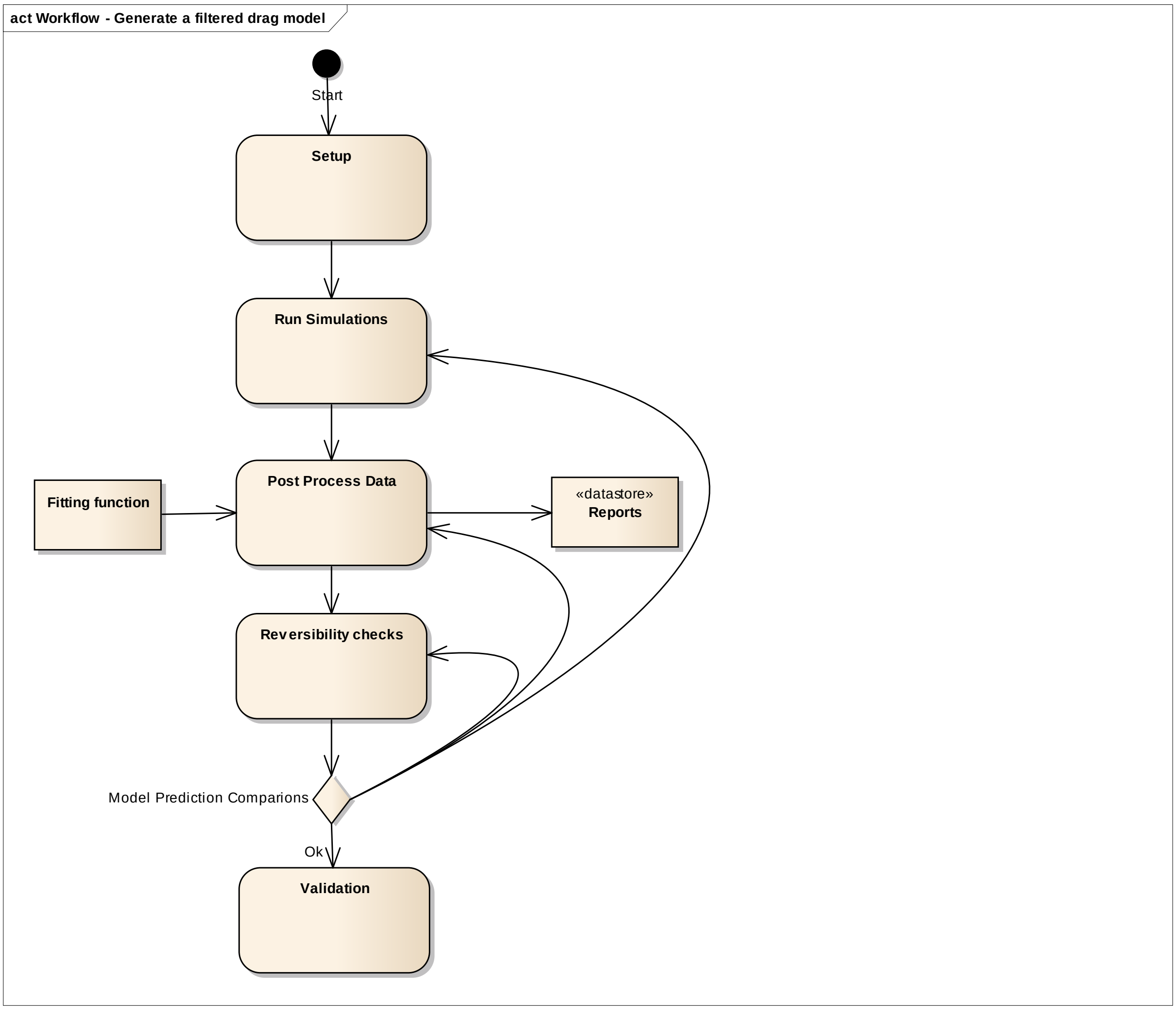

Figure 2 Activity Diagram - Generate a Filtered (drag) Model.

3.2 Quantum Mechanical Computation of Rate and Diffusion Constants

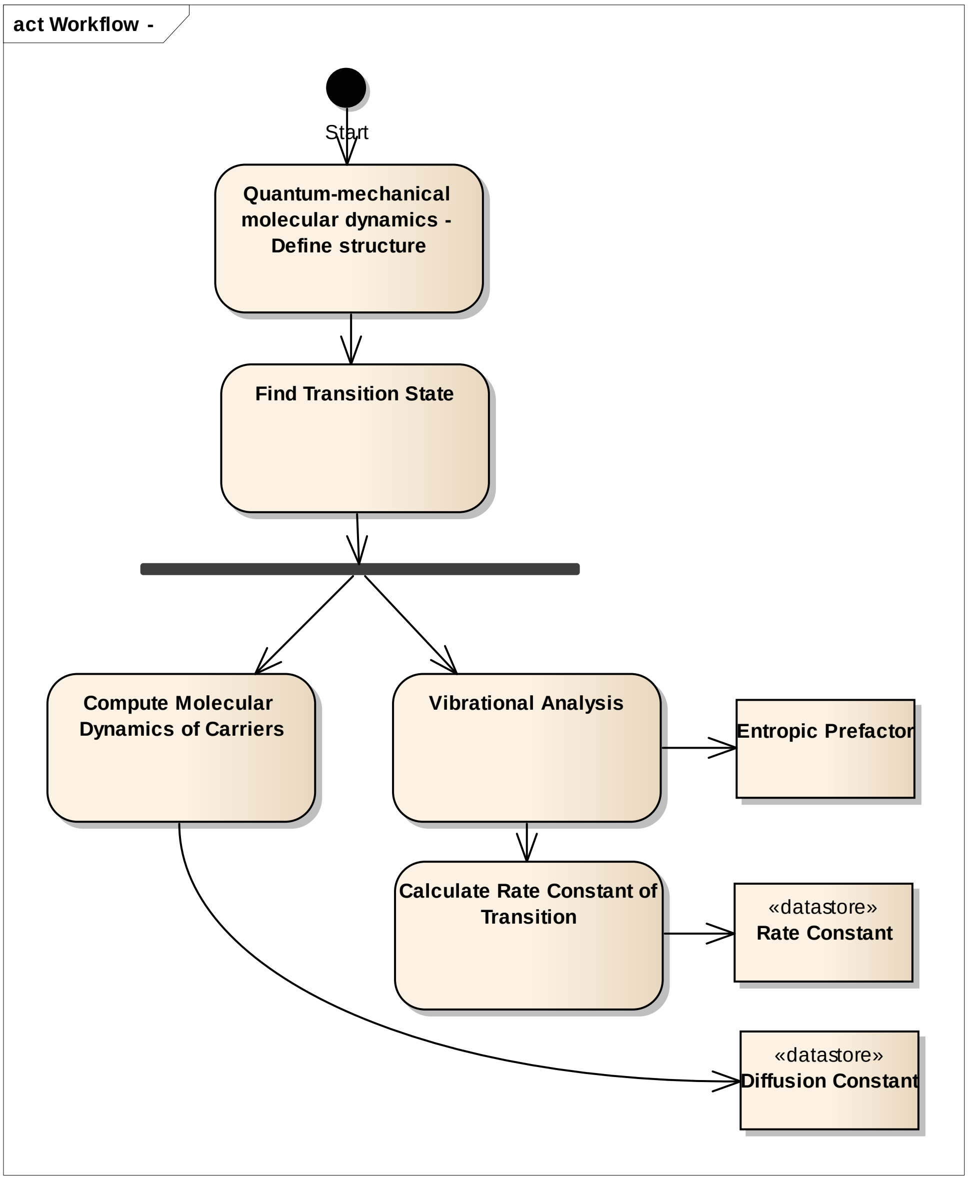

Figure 3 Activity Diagram - Quantum Mechanical computation of rate and diffusion constants

3.3 Kinetic Monte Carlo Simulation

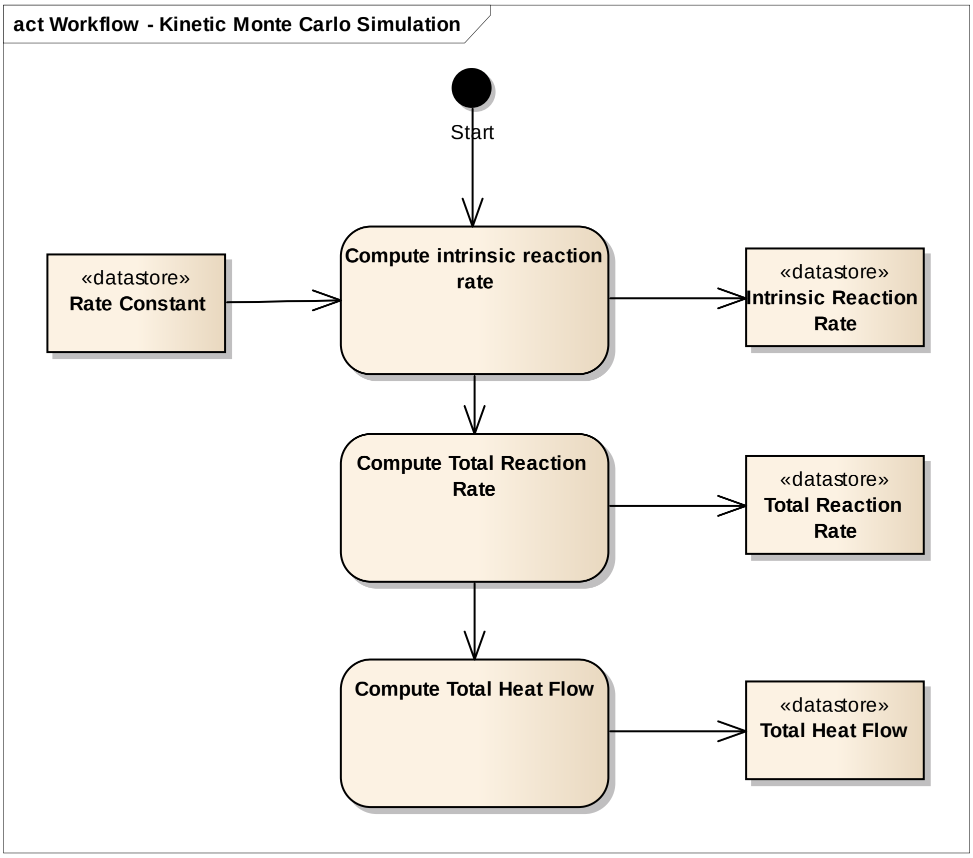

Figure 4 Activity Diagram - Kinetic Monte Carlo Simulation

3.4 Chemical Evolution of a Material

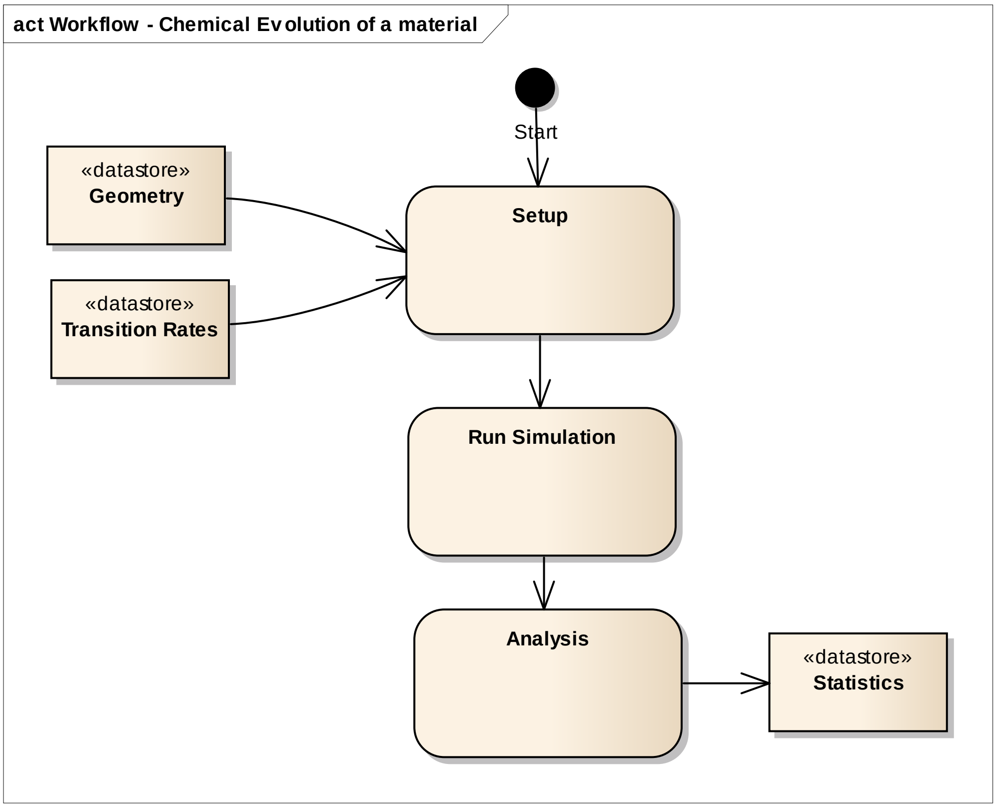

Figure 5 Activity Diagram - Chemical Evolution of a Material

3.5 Simulate a Lab-Scale Reactor with Complex Degassing Processes

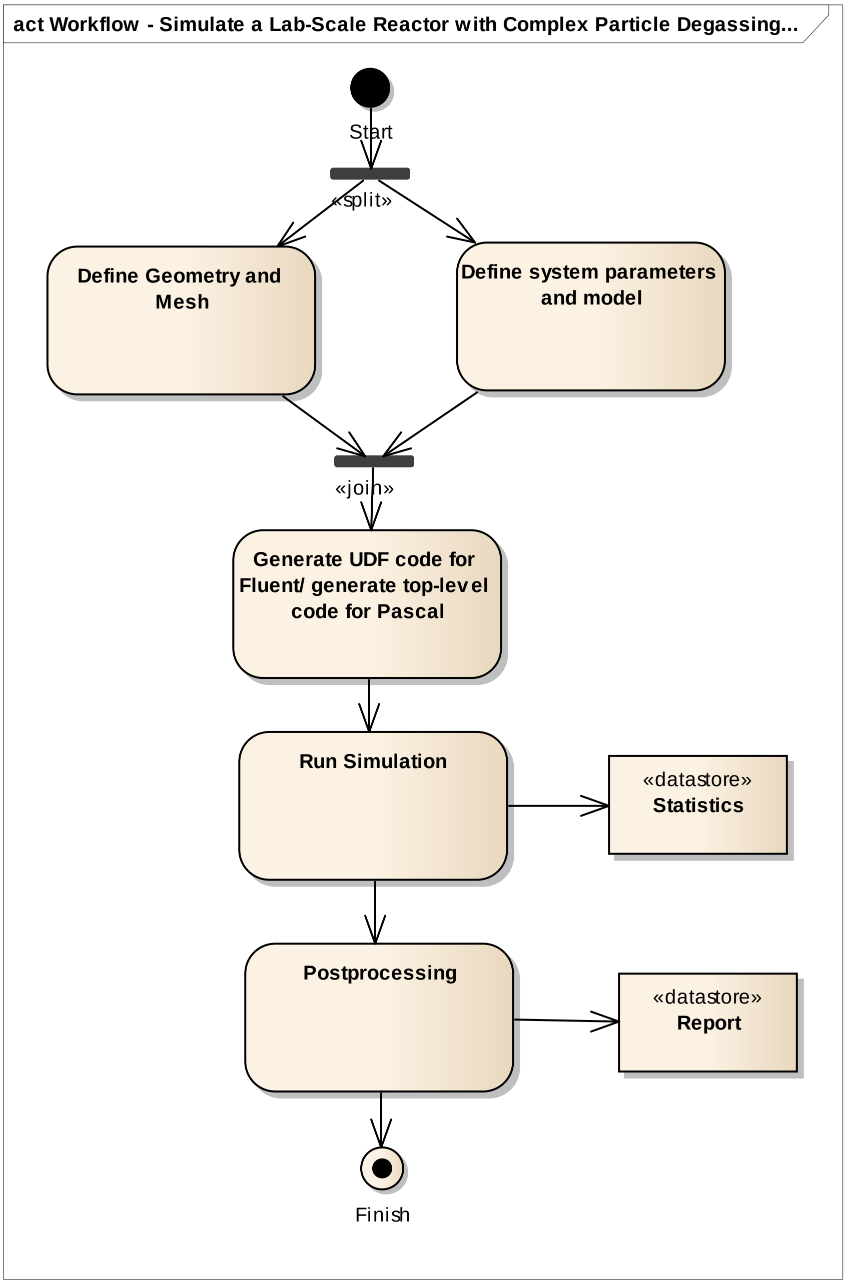

Figure 6 Activity Diagram - Lab Scale Reactor Simulation

3.6 Metal Distribution in a Porous Particle Prediction

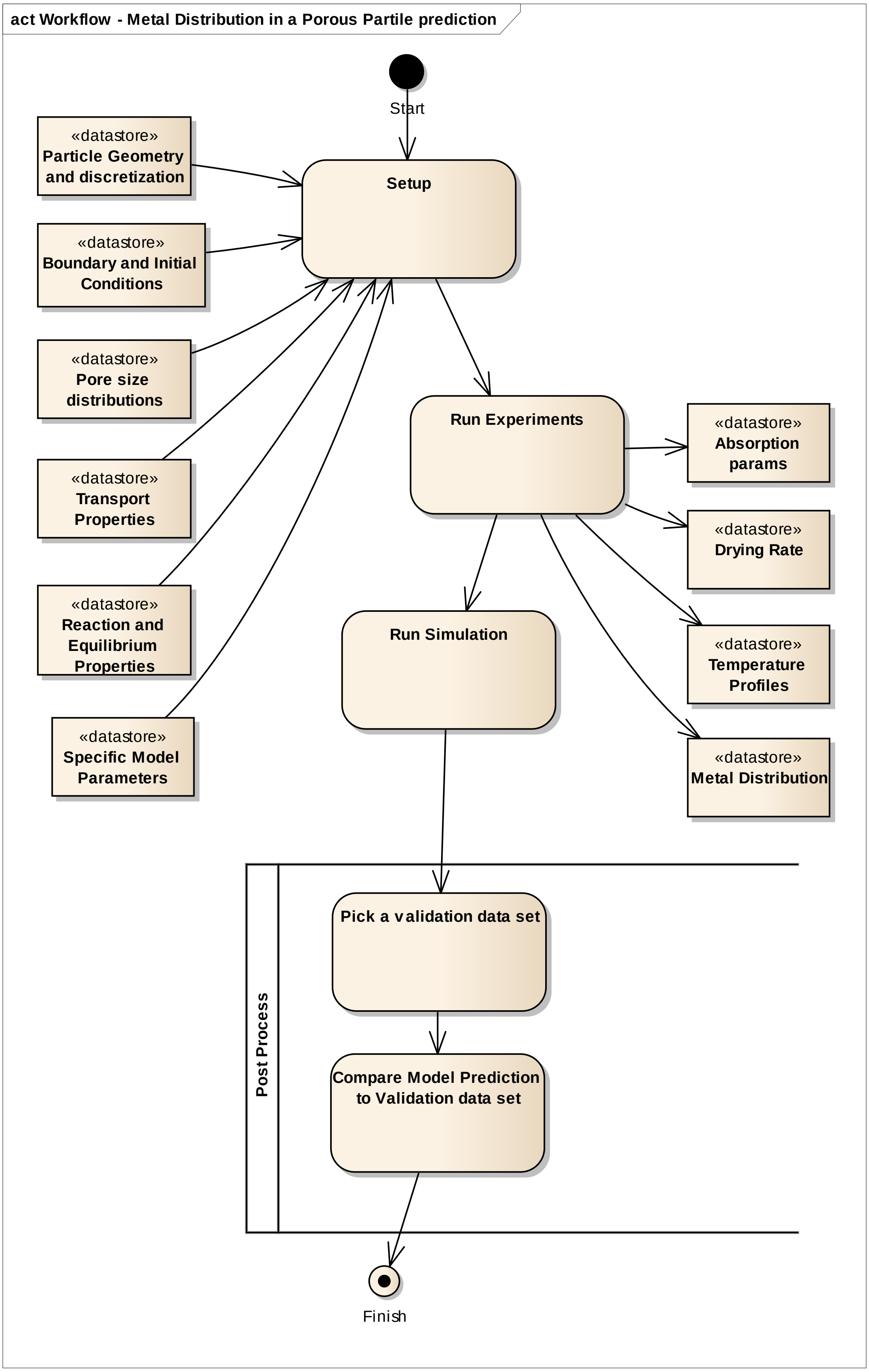

Figure 7 Activity Diagram - Prediction of Metal Distribution in a Porous Particle