COUPLING WEAP AND LEAP MODELS USING INTERACTION MODELING

Mostafa D. Fard
Hessam S. Sarjoughian

Arizona Center for Integrative Modeling & Simulation
School of Computing, Informatics, and Decision Systems Engineering
Arizona State University
Tempe, AZ, USA
{smd.fard,sarjoughian}@asu.edu

ABSTRACT

It is useful to model the interactions between the water and energy models separately. A coupled water and energy model using an interaction model lends itself to developing flexible, yet rigorous predictive simulation studies. A WEAP-LEAP interaction model is developed based on the Knowledge Interchange Broker modeling approach. A RESTful simulation framework is developed for coupling the Componentized-WEAP and Componentized-LEAP systems. Every interaction model is defined to have a set of modules where each module has its input and output ports with data transformation schemes. The input and output ports of the Componentized-WEAP and Componentized-LEAP models are coupled with those of the interaction model. A cyclic, synchronous execution protocol is defined for concurrent, bidirectional data transformations. The computational efficiency of this loosely coupled water-energy framework is acceptable compared with the tightly WEAP-LEAP system. This proposed framework shows understanding the water-energy system nexus is advantageous using interaction modeling.

Keywords: Water-Energy Nexus, Component-based modeling, Knowledge Interchange Broker, WEAP, LEAP.

1 INTRODUCTION

Water and energy are essential resources to sustain the development of our society. By 2030, water demand is expected to increase by 40% (Addams, et al. 2009) and energy by 50% (Van der Hoeven 2013) concerning the 2010 levels, mainly due to the increasing population and higher standards of living in developing countries. The resources are often managed separately, although the policymakers and resource managers need to understand the relationship between them. It is essential for making informed and mutually compatible decisions for short-/long-term planning. Meeting these resource challenges requires integrated worldviews. The first step for integration is to improve our understanding of how these resources are linked and the degree to which they depend on each other. Consequently, combined water and energy resource planning is receiving more and more attention from varied stakeholders, including academia, the general public, and local, national, and international organizations. Thus, it is no surprise that an in-depth understanding of the Water-Energy nexus is a crucial toward sustainable resource planning.

Water-Energy nexus is rooted in bi-directional dependency. For example, water is needed to cool power plants, and energy is required to pump, treat, and distribute water. In addition to these internal interactions between the systems, other external factors (e.g., climate change, population growth, and economic instabilities) increase the complexity of the Water-Energy system. For example, what will happen in the

nexus if the price of water increases by a resource manager? Studies approaching the Water-Energy nexus have been presented using various methods, tools, and frameworks (Hamiche, Boudghene Stambouli and Flazi 2016, Khan, et al. 2018, Dai, et al. 2018).

Considering the water and energy systems as separate models and coupling them via a third model brings up not only the modularity for the systems but also flexibility and rigorous predictive simulation for the integrated Water-Energy nexus. Therefore, the three models can be considered to represent the whole system; one for the water system, one for the energy system, and one for their nexus. From a high abstract modular perspective, water and energy systems have some input/output ports to interchange data from/to the other system. In a general view, systems can have different model structures and behaviors defined according to formal specifications. This research is based on using the Water Evaluation and Planning (WEAP) system (WEAP 2020) for the water system and the Long-range Energy Alternatives Planning (LEAP) system (LEAP 2020) for the energy section. The advantage of relying on previously established frameworks can be the reduction in the effort and resources needed for model development. Domain experts can participate in collective work using existing tools and can reduce learning curves and allow for better use of previously acquired knowledge and experience.

2 BACKGROUND

The WEAP system, the LEAP system, the Componentized-WEAP (C-WEAP) framework (Fard and Sarjoughian 2019), and the Componentized-LEAP (C-LEAP) framework are briefly described next. These descriptions are to put in the perspective using these systems together, thus modeling and simulating water and energy systems as one system.

2.1 Water Evaluation and Planning System & Long-range Energy Alternatives Planning System

The WEAP system is a modeling, simulation, and evaluation tool for studying water supply and demand (WEAP 2020, Yates, et al. 2005). A water system can be modeled using predefined entities (Nodes and Links). The LEAP system is a policy analysis and climate change mitigation assessment for energy (LEAP 2020). An energy system can be modeled using four types of entities (Demand, Transformation, Process, Resource, and Effect). The WEAP and LEAP systems are using a scenario-based approach. Every scenario is a self-consistent storyline of how a system might evolve for a finite time period under some policy condition. Both systems create and/or parameterize the input variables and equations for the defined entities. The WEAP and LEAP tools are widely used by domain experts and researchers to model the water and energy systems (Howells, et al. 2013).

From the simulation perspective, the WEAP and LEAP systems are based on DTSS (Discrete Time System Specification), and they have an uninterruptible execution. All input data must be ready before the start of every simulation. All output data will be accessible after a whole simulation execution period is completed. In this respect, the WEAP and LEAP models are not reactive since they cannot have input from any external simulation model while being executed. The WEAP and LEAP systems are closed-source and developed using the Delphi programming language (Sieber 2005). Even though the entities with their input and output variables are known, their mass-balance equations are not known and cannot be discovered from outside. The variables for the entities appear to be shared within the WEAP and LEAP tools. A water/energy model's logical and schematic parts are tightly interwoven to the scenarios, configurations, and results. Both tools have predefined Application Programming Interfaces (APIs) to access most of the entities and execution control. The time granularity is one of the significant concepts in the WEAP and LEAP systems. The time-step in the WEAP system divides a year to equal segments, for example, yearly (one step per year), monthly (12 steps per year), daily (365 steps per year). The same concept exists in the LEAP system called time-slice, but time-slices must be defined one-by-one by the user, and the size of the segments can be different. For example, the first time-slice can cover the first three months of a year, and the next time-slice can cover the rest of the year (9 months), but time-slices must cover the whole year.

2.2 Componentized-WEAP & Componentized-LEAP Frameworks

The C-WEAP (Fard and Sarjoughian 2019) and C-LEAP frameworks are two model components replicated externally to the WEAP and LEAP systems. They allow using the systems as a component-based tool supported within a RESTful framework. These frameworks have model components for all entities that are defined in the WEAP and LEAP systems. Using the frameworks, the project, and different types of components (Node, Link, and Flow for the WEAP system, and Resource, Transformation, and Demand for the LEAP system) in a model and their data are accessible externally via the RESTful APIs. The returned data via different APIs is in JSON format (correspond to the defined domain models for the components and variables). The C-WEAP and C-LEAP frameworks present the same schema for a project, its scenarios, components, and the input/output variables belong to each component.

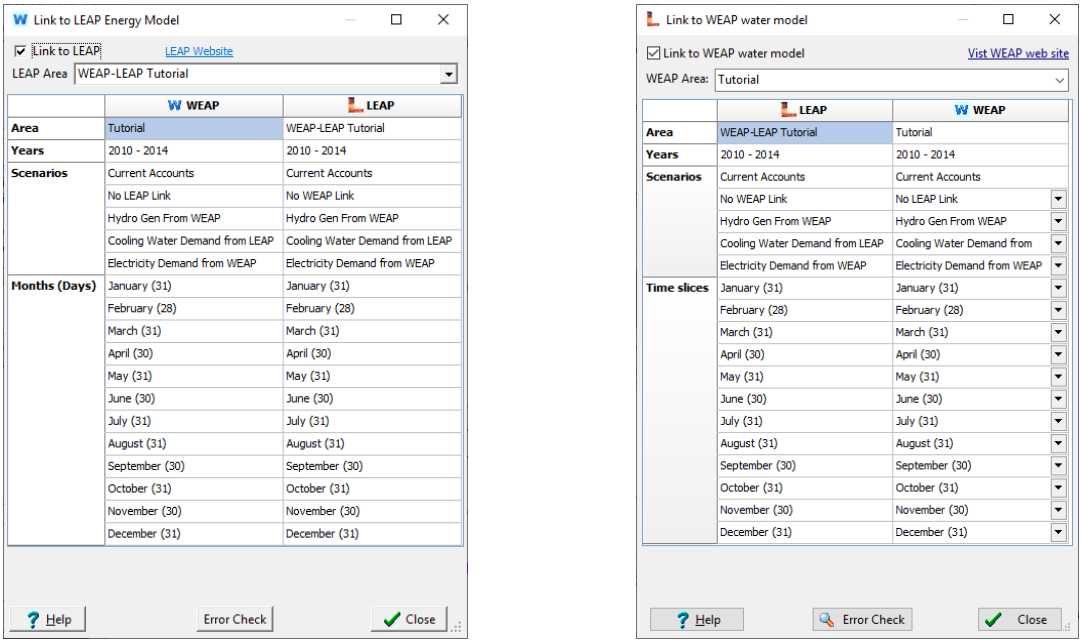
3 RELATED WORK

Many component-based modeling frameworks have been developed in recent years. A system can be defined as a collection of interacting components in a framework and implemented in object-based or object-oriented programming languages. In contrast, a framework based on separate data structure and function abstractions can be used to model a system and implement it in procedural programming languages. The WEAP and LEAP systems fall to the latter modeling framework. Considering simulation studies of the FEW Nexus, frameworks and tools such as WEF (Daher and Mohtar 2015) are not founded on component-based modeling principles. The WEF Nexus Tool 2.0 (Daher and Mohtar 2015) is a scenario-based tool for guiding resource allocation at the country level for a given level of food self-sufficiency and a set of technologies, land uses, and resource availabilities. Some other frameworks, such as NexSym (Martinez-Hernandez, Leach and Yang 2017), CLEWS (Howells, et al. 2013), and PRIMA (Kraucunas, et al. 215), have some modular approaches for the FEW. The NexSym advances in nexus tools by explicit dynamic modeling of local techno-ecological interactions relevant to WEF operations. The modular tool integrates models for ecosystems, WEF production, and consumption components and allows the user to build, simulate, and analyze a "flowsheet" of a local system. The CLEWS framework integrates existing simulation tools (WEAP, LEAP, and AEZ) using a modular structure that allows analyzing interactions between interconnected sectors. The PRIMA features enhanced domestic resolution, individual sector models, and a flexible, portable, and modular platform that facilitates its application to different types of analyses, regions, and components.

From a model coupling perspective, the WEAP and LEAP systems use a hidden internal linking mechanism to interact with one another. As shown in Figure 1, in each system, a table is displayed in a form to specify mappings between the WEAP and LEAP scenarios and time granularities for given projects. Two restrictions must be satisfied for this connectivity. First, both projects must have the same start and end years for a simulation. Second, both projects must have a matching set of time-steps (e.g., monthly, and daily); otherwise, some data will be lost. After establishing the connection, a system has access (via the equations/expressions) to read the Data and Result variables of different entities from the other system.

4 APPROACH

The Knowledge Interchange Broker (KIB) approach has been introduced to formalize the interactions between the models specified in different modeling formalisms (Sarjoughian 2006). The KIB formalism can be used to define data mappings, synchronization, concurrency, and timing. The conceptual basis of the KIB is that disparities between different syntaxes and semantics need to be accounted for with a separate model syntax and semantics, thus enabling independent modeling of interactions between the composed models. This approach has been applied to different domains (Huang, et al. 2009, Barton, et al. 2016). In this research, the KIB concept is used to define externally define the relationship between the WEAP and LEAP discrete-time models. The aim is to define the water and energy nexus as a separate model.
An interaction model composing the water and energy models is developed based on the KIB approach. The architecture illustrating the abstract specification of the nexus (i.e., interaction) between the WEAP an-



(a) (b)

Figure 1: The WEAP-LEAP internal linking mapping configuration. (a) Linking an energy model to a water model. (b) Linking a water model to an energy model.

d LEAP systems is shown in Figure 2. Choices of the water and energy model components and the constraint under which their data can be transformed for use by one another are defined in the Data Transformation part of the interaction model. The data sets in the water and energy models are used to define a set of modules (data transformation schema) that can be executed under a time-based control regime. An execution protocol prescribes a cyclic control regime supported by synchronous calls and returns enabled by RESTful web services and JSON. A simple time management scheme is devised to synchronize different time resolutions that can be used in the water and energy models. The interaction model has interfaces of the C-WEAP and C-LEAP frameworks. The WEAP Interface is defined as a part of the interaction model for bi-directional, synchronous communication with the C-WEAP models. The WEAP Interface mediates communication between the RESTful web services of the C-WEAP, and the modules in the interaction model. In response, the C-WEAP returns the data (JSON) from the water system to the WEAP Interface. The LEAP Interface is defined in the same way as the WEAP Interface.

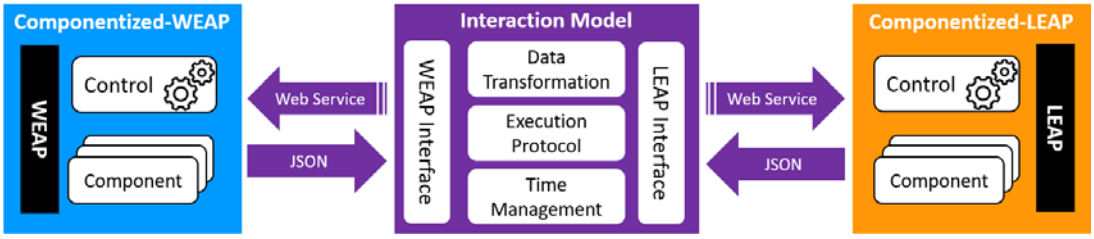


Figure 2: A Water-Energy Nexus high-level architecture.

4.1 WEAP-LEAP Interaction Model Specification

The UML class diagrams for the Water-Energy nexus interaction model are included in the `core` and `WEAP-LEAP` package diagrams, as shown in Figure 3 and Figure 4. The `core` package diagram has the concrete classes `IM`, `Coupling`, `TransformationInputPort`, and `TransformationOutputPort` that must be instantiated as well as the interface `IMessage` that must be implemented. The abstract classes `System`, `Module`, `ModuleInputPort`, and `ModuleOutputPort` must be inherited for a specific coupled Water-Energy model. Following the internally integrated WEAP and LEAP system execution mechanism (WEAP 2020,

Yates, et al. 2005), the interaction model executes any two composed water and energy models in a round-based fashion. The `UUID` (Universally Unique Identifier) datatype is used for the `id` attribute in the `Element` abstract class. The `ComponentTypes` in the `Component` abstract class is either the `Module` or `Transformation` enumerated datatype. Also, the `PortTypes` in the `Port` abstract class is either the `Input` or `Output` enumerated datatype. For brevity, the setter and getter operations are excluded from the classes shown in the package diagrams.

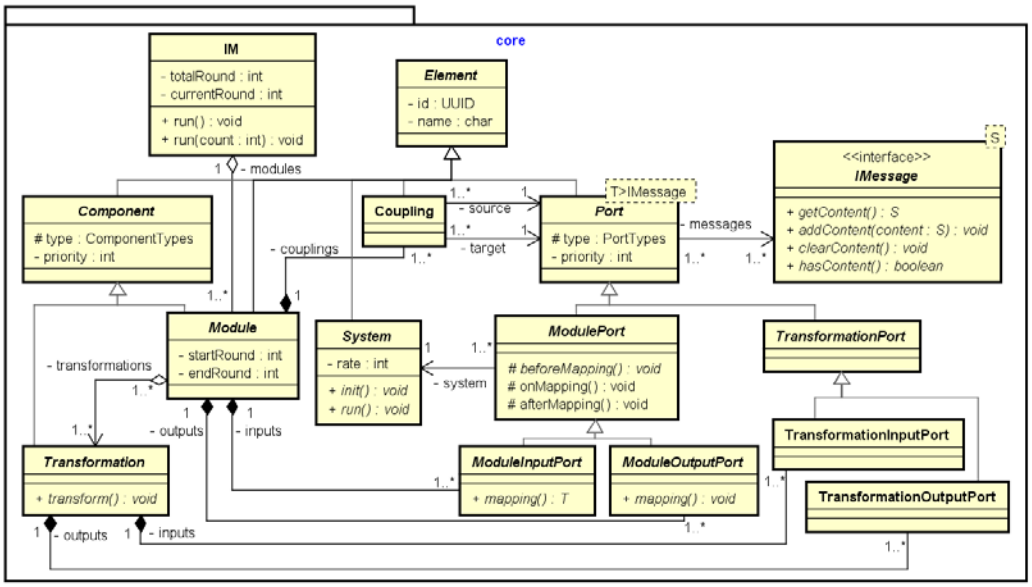


Figure 3: The core package for the Water-Energy nexus model.

The `WEAP` and `LEAP` classes in Figure 4 are proxies for the C-WEAP and C-LEAP frameworks. The module input ports `InputFromWEAP` and `InputFromLEAP` and the module output ports `OutputToWEAP` and `OutputToLEAP` are defined for the connected ports to the C-WEAP and C-LEAP frameworks. According to (Fard and Sarjoughian 2019), some attributes (`projName`, `compType`, `compName`, etc.) are defined to make the APIs for the related systems (attributes for these classes are the same). The module ports are entirely independent of each other, and they can have different attributes and operations. The `WEAPMessage` and `LEAPMessage`, implement the `IMessage` interface, are defined as structures for the incoming/outgoing messages from/to the C-WEAP and C-LEAP frameworks (see Figure 4). Each `WEAPMessage/LEAPMessage` has a finite number of time intervals, subject to the constraints of the frameworks, with each time interval having a time-step/time-slice and a value. As an example, two modules (*Module1* and *Module2*) and two transformations (*Transformation1* and *Transformation2*) are defined in the `WEAP-LEAP` package diagram (see Figure 4). Descriptions for the class and package diagrams are excluded due to space limitations.

The Interaction Model interacts with external systems using the inputs and outputs defined for its modules and the transformations in them. The Interaction Model communicates with the RESTful C-WEAP and C-LEAP frameworks. The input and output ports for the modules are independent as well as those that are defined for the transformations. All connections within the interaction model are uni-directional. In Figure 5, the cloud shape represents the webserver with the APIs defined for the C-WEAP and C-LEAP. The APIs are depicted as circles in the cloud. Each component is shown as a rounded rectangle with its input and output ports shown as arrows. Multiple APIs can communicate with a component. Each API can read the inputs or outputs data of a component or apply some changes to the inputs. As an example, the URL "`/Water/demo/DemandSite/phoenix/Input/Annual%20Activity%20Level/Reference`" accesses the data of the *Annual Activity Level* input variable of the *phoenix* demand site of the *Reference* scenario in the *demo* project of the WEAP system. The relevant URLs to call APIs from the C-WEAP/C-LEAP frameworks are defined in the `InputFromWEAP/InputFromLEAP` classes (see Figure 4). The component, port, and scenario data are used to retrieve data from the systems. The `OutputToWEAP` and `OutputToLEAP`

classes are defined for writing data to the systems. The APIs calls are defined in the `mapping()` method of the classes inherited from the `ModuleInputPort` and `ModuleOutputPort` classes (see Figure 3 and 4).

All data transformation specifications must conform to certain constraints. The name of each element in its content must be unique. Examples include the module's name in the Interaction Model content, the module input/output port's name in a module content, the transformation's name in a module content, and the transformation input/output port's name in a transformation content. Also, there are three valid types of coupling (connections between two ports). First, coupling from a module's input port to a transformation's input port. Second, coupling from a transformation's output port to a module's output port. Third, coupling from a transformation's output port to another transformation's input port. Thus, it is not valid to have a self-coupling for any transformation. Figure 5 shows the schema of a valid data transformation for the Water-Energy nexus. This IM contains one module (*Module_1*) with two input ports (*In1* and *In2*) and two output ports (*Out1* and *Out2*). These ports are linked to the C-WEAP and C-LEAP systems. The module contains two transformations, *Transformation_1* with two input ports (*In1* and *In2*) and two output ports (*Out1* and *Out2*), and *Transformation_2* with one input port (*In1*) and one output port (*Out1*). The module has six couplings, three couplings from module input ports to the transformation input ports (e.g., from *Module_1.In1* to *Tranaformation_1.In1*), two couplings from transformation output ports to the module output ports (e.g., from *Transformation_2.Out1* to *Module_1.Out2*), and one coupling from the transformation output port to the transformation input port (e.g., from *Transformation_1.Out2* to *Tranaformation_2.In1*). The structure of incoming or outgoing data on a port (the class implemented the `IMessage` class) cannot change during execution. Each module or transformation port can accept a specific message type. As a result, the source and the target of every coupling must have the same message structure.

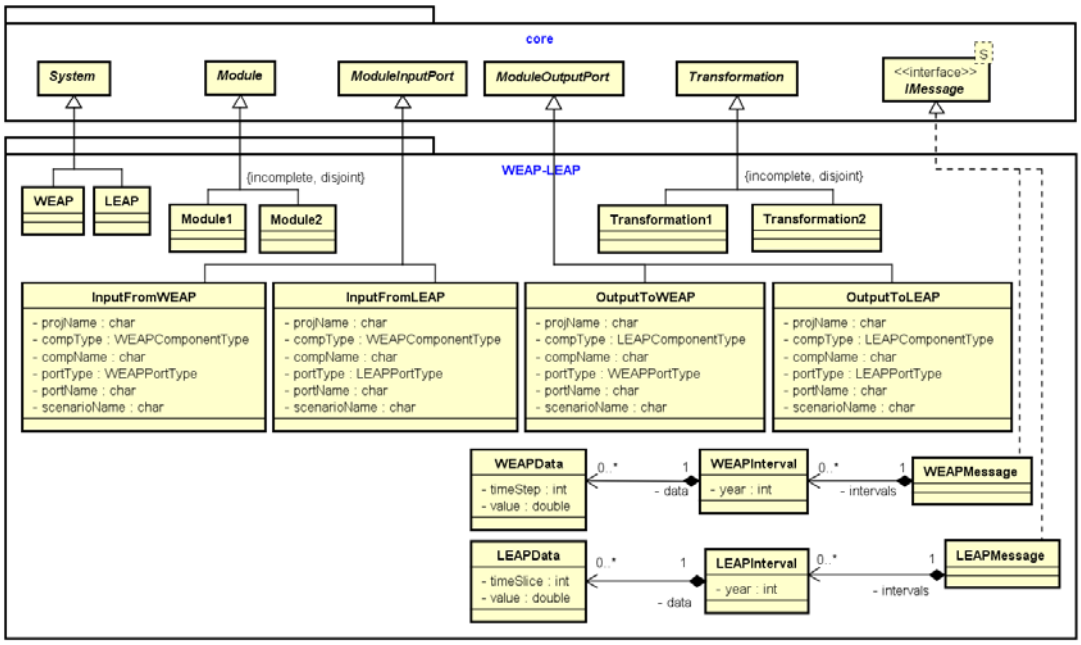


Figure 4: WEAP-LEAP Interaction Model class diagram having two modules and two transformations.

4.2 Coupled WEAP-LEAP Execution Protocol

The execution engine for the Water-Energy nexus models has a round-based execution for the C-WEAP and C-LEAP systems. Each complete round consists of six steps, as shown in Figure 5. In the first step, the connected systems to the module's input ports are executed. In the second step, the data are invoked by module input ports from the systems. In the third step, the received data are sent to the transformation input ports. In the fourth step, the transformation units are executed (to transform input data to the output data). In the fifth step, the processed data are sent to the module's output ports/other transformation input ports.

In the sixth step, the output data (collected in the module's output ports) are sent to the systems. Both systems independently and simultaneously execute.

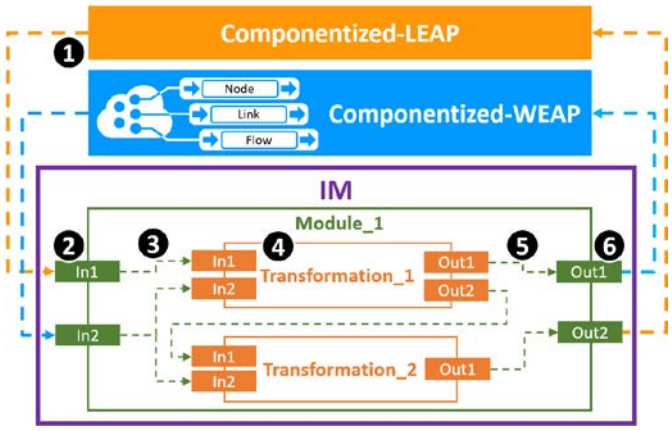


Figure 5: An illustration of the data transformation process for the coupled Water-Energy system.

The algorithm shown in Listing 1 is used to execute every Water-Energy nexus interaction model based on the six steps in Figure 5. In lines 2-3, the $total\text{-}round$ and $current\text{-}round$ variables are set to N and 0, respectively. In line 4, the S is the set of the C-WEAP and C-LEAP systems. In lines 5-6, the $init()$ function of all linked systems to the Interaction Model is executed. According to the initial values for $total\text{-}round$ and $current\text{-}round$, the body of the while section (lines 8-38) will be run for N iterations. In the first step (lines 8-10), the $run()$ function of the active systems are executed. A system is active when the modulo of the $current\text{-}round$ to its $rate$ (i.e., the number of steps in a time period) equals to zero (line 9). Also, a module can be *active* if the $startRound$ is smaller than or equals to the $current\text{-}round$ and the $endRound$ is bigger than or equals to the $current\text{-}round$ (line 11). In the second step (lines 12-15), the $mapping()$ function of the module's input ports of active modules that are connected to the active systems is executed. In the third step (lines 16-22), the data in the module's input ports (if there is data) are transferred via couplings to the transformations input ports. If a module input port is coupled to multiple transformation's input ports (e.g., module input port "*In 2*" in Figure 5), the data is copied to all destinations (lines 19-21), then the module's input port is vacated (line 22). In the fourth step (lines 23-25), the $transform()$ function is applied to each transformation of all active module (i.e., all input data are processed and the resultant data are sent to the transformations' output ports). In the fifth step (line 26-33), the data in the transformations' output ports are transferred via couplings to the transformations' input ports/module's output ports. Again, if a transformation output port is coupled to multiple destination ports, the data is copied to all the receiving transformations in the module (lines 30-32) and then the transformations' output ports are vacated (line 33). The $current\text{-}round$ value is increased by one unit in line 34. Finally, in the sixth step (lines 35-38), the $mapping()$ function of the module's output ports of active modules that are connected to the active systems are executed to send data to the systems. It is noted that all modules execute independently of one another. For brevity, the sorting operation for modules, transformations, and ports are not shown. For example, modules are sorted by priority in line 11 and module input ports are sorted by priority in line 13.

4.3 Coupled WEAP-LEAP Time Management

The time resolution for coupled WEAP and LEAP models uses the timing specification defined in the Water-Energy nexus model. The `rate` property in the `System` class (see Figure 3) and the execution protocol (see Section 4.2) synchronize the WEAP and LEAP models. The classes implementing the `IMessage` interface and the defined transformations (see Figure 3) control matching the time intervals defined for the water and energy models in each execution round. The `rate` property of the `System` class in Figure 3 defines the ratio of running the system related to the round of the `IM`. For example, if the time interval for a water model is a year (e.g., the start year is 2020 and the end year is 2021) and the time interval

Algorithm 1 Run Simulation

```
1: procedure RUN()
2:     total-round ← N
3:     current-round ← 0
4:     S ← distinct Systems connected to the input & output ports of all modules in the IM
5:     for each (s in S) do
6:         s.init()
7:     while (current-round < total-round) do
8:         for each (s in S_I) do                ▷ S_I ⊆ S, systems connected to the module's input ports
9:             if (current-round % s.rate == 0) then
10:                s.run()
11:        active-modules ← IM.modules that (m.startRound ≤ current-round ≤ m.endRound)
12:        for each (m in active-modules) do
13:            for each (p in m.inputs) do
14:                if (current-round % p.system.rate == 0) then
15:                    p.mapping()
16:        for each (m in active-modules) do
17:            for each (p in m.inputs) do
18:                if (p.messages.size() > 0) then
19:                    for each (c in m.couplings) do
20:                        if (c.source == p) then
21:                            c.target.messages ← c.source.messages
22:                    p.messages ← φ
23:        for each (m in active-modules) do
24:            for each (t in m.transformations) do
25:                t.transform()
26:        for each (m in active-modules) do
27:            for each (t in m.transformations) do
28:                for each (p in t.outputs) do
29:                    if (p.messages.size() > 0) then
30:                        for each (c in m.couplings) do
31:                            if (c.source == p) then
32:                                c.target.messages ← c.source.messages
33:                        p.messages ← φ
34:        current-round ← current-round + 1
35:        for each (m in active-modules) do
36:            for each (p in m.outputs) do
37:                if ((current-round % p.system.rate == 0) & (p.messages.size() > 0)) then
38:                    p.mapping()
```

Listing 1. A sketch of the Interaction Model execution protocol.

for an energy model is ten years (e.g., the start year is 2020 and the end year is 2029), then the rate of a `WEAP` must be set to one, and the rate of the `LEAP` must be set to 10 in the Interaction Model. For a coupled WEAP-LEAP model, the time period for the WEAP is given for the first year with all subsequent years to be simulated. In this scenario, for every 10 execution cycles of the WEAP model, there is 1 execution cycle of the LEAP model. In this configuration, the Interaction Model executes 10 times per 1 execution cycle of the LEAP model.

Finer-grain time resolution can be defined using the classes implementing the `IMessage` interface. For example, a WEAP or LEAP model can have a finite discrete-time resolution for a year (Fard and Sarjoughian 2019). The time resolutions for the data transformations from the WEAP to the LEAP (or vice versa) follows the restrictions provided in the WEAP and LEAP systems, individually and together. The

time-values relevant to the classes realized for the `IMessage` interface (the `WEAPMessage` and `LEAPMessage` in Figure 4) can be mapped to some other values by aggregating and/or disaggregating data in the water and energy models. The modeler can define time interval conversions for each data transformation consistent with the time intervals used in the water and energy models.

5 WEAP-LEAP INTERNAL LINKING

The WEAP and LEAP systems can bi-directionally share data with one another (refer to Section 2.1). Each system can read the Data and Result variables of different entities from the other system. The path to an entity and a specific variable must be defined via form wizards (in the tree view for the entities and the list view for the variables) or writing the expression via text wizards. Paths are difficult to define using wizards when the user wants to manipulate more than one variable from one other system for a variable in another system. The user must know the details of the models in the WEAP and LEAP systems. Using the wizards become increasingly more challenging as the scales of the water and energy models increase. The *"WEAPValue(entityPath:variableName[unitName])"* function needs to be defined in a LEAP model for accessing a variable in the WEAP model. The *"LEAPValue(entityPath:variableName[unitName])"* function needs to be defined in a WEAP model in the same way for accessing a variable in the LEAP model. All algebraic equations, some of which can be complex, must be defined in the expression part of the variable of the destination model. Defining a variable in a water model to use variables from multiple entities in an energy model is cumbersome. Defining paths between an energy model with many variables that use many variables in a water model becomes complicated and error prone.

The variables in the WEAP and LEAP models can be mapped to one another using the time resolution types I, II, and III shown in Figure 6. The arrow directions in the figures signify the data of a target time-step/time-slice is read from the data of a source one. Also, it is supposed that the first time-step is mapped to the first time-slice, the second time-step to the second time-slice, and so on in the WEAP-LEAP internal linking form (see Figure 1). For the Type I time resolution, both variables have the same number of time-steps and time-slices in the WEAP and LEAP systems, respectively. As shown in Figure 6(a), the time resolution for the water and energy models are the same. For the Type II time resolution, one of the variables (from the WEAP or LEAP) has a smaller time resolution, and the variable with the smaller time resolution reads the variable with a larger time resolution from the other system. In Figure 6(b), the variable X in the LEAP model with yearly time-slice is read by the variable Y in the WEAP model with n time-steps. As a result, the x_1 variable is used by all time-steps for values $y_1, y_2, \dots, y_n$ of variable Y. From a yearly perspective, the value x_1 which is for the whole year of the variable X in the LEAP system is multiplied by n for variable Y in the WEAP system. This problem can be solved by applying a distribution on the smaller time resolution. For example, by dividing the x_1 by n on the WEAP side in Figure 6(b) for uniform distribution. For the Type III time resolution, one of the variables (from the WEAP or LEAP) has a smaller time resolution, and the variable with smaller time resolution is read by the other variable. In Figure 6(c), the variable Y in the WEAP system with n time-steps is read by the variable X in the LEAP system with yearly time-slice. The value of the first time-step of variable Y is read by the variable X and the remaining values y_i $(i = 2, \dots, n)$ are ignored. This restriction cannot be removed using the WEAP-LEAP internal linking, and some data is lost. A well-defined specification is presented in the proposed coupled WEAP-LEAP framework for all types of interactions in the WEAP-LEAP internal linkage (based on the time resolutions).

In the WEAP-LEAP internal linking, the modeler must decide and manually execute data exchanges between the water and energy models. After the WEAP-LEAP internal linking, the execution of each system depends on the execution of the other one, because a system may generate some data (as the output) which are used by the other system. All the input data for the whole simulation interval (from start year to the end year) must be ready before simulation execution in the WEAP or LEAP. Also, all output results are generated after the simulation execution (before the execution, they are zero). Consider a situation where each model reads the output of the other model as its input. Initially, both models have some data as their inputs and zero values as their outputs. Then, the modeler must manually execute either the water or energy

model, first. Consequently, the first model reads zero values (outputs) from the second model and generates its own results. Then, the second model is executed and read the outputs of the other system (which can be non-zero values). In this proposed coupled WEAP-LEAP framework, the order of data transformations is defined in the interaction model and supported with automated execution.

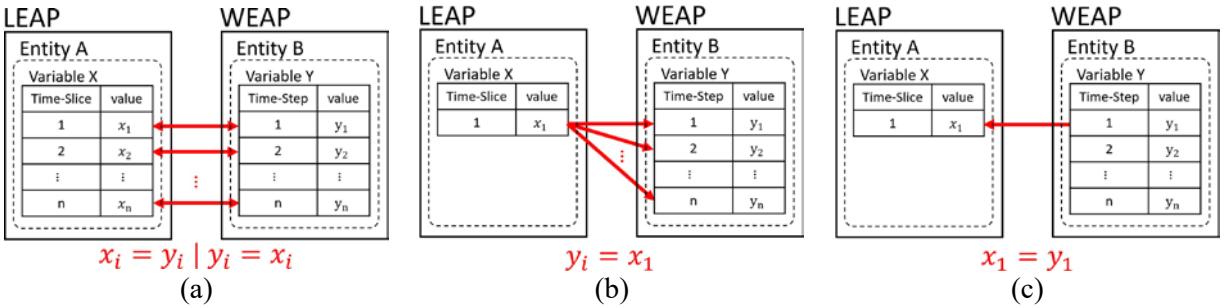


Figure 6: The interaction types in the WEAP-LEAP internal linking. (a) Type I: same time resolution. (b) Type II: larger to smaller time resolution. (c) Type III: smaller to larger time resolution.

6 DISCUSSION

It is useful to compare the WEAP-LEAP internal linking and the coupled WEAP-LEAP framework using Interaction Modeling. From the *Composition Model* perspective, the WEAP-LEAP internal linking has certain restrictions, particularly in terms of defining the exchange of data between water and energy models as a standalone interaction model. Using interaction modeling, individual data transformations each having its data type, data dis/aggregation calculations, and bi-directional exchange frequency (time resolution). The coupled WEAP-LEAP framework affords a high-degree of managing flexible data transformation interactions and control. The WEAP-LEAP internal linking supports connections via pre-defined input/output data sharing (see Figure 1) and equations. In comparison, each uni-directional module in the interaction models can have one or more data transformations specifying the nexus of coupled water and energy models. The modules in an interaction model together support independent, bi-directional interactions between the water and energy models. Also, the complexity of the mathematical calculation (which is supported via text expressions in the WEAP-LEAP internal linking) can be handled in the transformation of the interaction model. From the execution perspective, the WEAP-LEAP internal linking executes very fast compared with the coupled WEAP-LEAP framework.

To demonstrate a full simulation cycle using the Interaction Model, the *"Weaping River Basin"* project in the WEAP system and the *"Freedonia"* project in the LEAP system are used. The Interaction Model has one module with two transformations. The first transformation has two input ports and one output port. The input ports receive the *"Water Demand"* values (that is an output variable from the WEAP system) from the *"Industry North"* and *"Industry East"* demand sites entities in the WEAP system, and the output port of the transformation generates the result for the *"Resource Imports"* (that is an input variable of the LEAP system) of the *"Wind"* resource entity in the LEAP system. The second transformation has one input port and one output port. The input port receives the *"Activity Level"* values (that is an output variable from the LEAP system) from the *"Household"* demand entity in the LEAP system, and the output port of the transformation generates the result for the *"Variable Operating Costs"* (that is an input variable of the WEAP system) of the *"Industry North"* demand site entity in the WEAP system. Simulation time periods for steps one through sixth for one complete round (see Figure 5 and Listing 1) are 6.194, 1.145, 0.008, 1.114, 0.011, 1.063 seconds, respectively. In this example, the simulation time period accounting for one round of interaction from WEAP to LEAP and vice versa ranges 3 to 4 seconds on a desktop computer with 20 GB RAM and Intel Core i5 3.2 GHz CPU on Windows 10 64-bit.

Considering multi-processor or distributed simulation of a WEAP-LEAP model, the internal linking configuration (see Figure 1) requires both the WEAP and LEAP tools to execute on a single computer. In the coupled framework, each system (WEAP via C-WEAP, LEAP via C-LEAP, and IM) can be executed

on separate computing logically distinct processors or physically distributed computing platforms via web services. As mentioned before, the WEAP-LEAP internal linking can support Type I (see Figure 6) interactions, whereas the coupled framework supports the three types of interaction. This framework affords to define time resolutions for the data transformations separately but restricted to the time resolutions allowed in the WEAP/LEAP systems. From the nexus modeling perspective, the data sharing in the WEAP-LEAP internal linking is replaced with I/O data/coupling, which affords the framework to be modular.

The C-WEAP and C-LEAP frameworks are implemented using the NodeJS and TypeScript frameworks. The main packages used in the frameworks are *TS-Node 8.4.1* (Typescript-Node) for Typescript in the NodeJS server-side application; *Express 4.17.1* is used to build a web application and APIs; *Routing-Controller 0.8.0* is used to create structured, declarative and organized class-based controllers; *Body-Parser 1.19.0* to parse the incoming request to web-server, and *Winax 1.12.0* is used to define *ActiveXObject* in NodeJS (create WEAP instance in server-side application). The Water-Energy interaction model is implemented using Java 8, and the Jersey (Kalin 2013) framework is used to call the RESTful web services for the C-WEAP and C-LEAP frameworks. The WEAP tool (version 2019.2) and the LEAP tool (version 2018.0.1.32) are used for the above demonstration example.

7 CONCLUSIONS

Addressing the needs to integrate tools for understanding and assessing the Water-Energy nexus, this research presents a new modeling and simulation framework, based on the Knowledge Interchange Broker approach for coupling models developed in the WEAP and LEAP systems. Even though these tools are internally linked, defining interactions between water and energy models is limited in terms of flexibly defining choices of data to be communicated, time resolution, and control with support for distributed simulation. This framework allows defining the nexus of the water and energy systems as a separate model using well-formed modules, transformations, and ports as compared to the closed WEAP-LEAP internal linking. As future work, the use of the interaction modeling is planned to be extended to support modeling and simulation of Food-Energy-Water systems-of-systems.

ACKNOWLEDGMENT

This research is funded by the National Science Foundation under Grant #CNS-1639227, “INFEWS/T2: Flexible Model Compositions and Visual Representations for Planning and Policy Decisions at the Sub-regional level of food-energy-water nexus”.

REFERENCES

Addams, L., G. Boccaletti, M. Kerlin, and M. Stuchtey. 2009. "Charting our Water Future: Economic Frameworks to Inform Decision-Making." *McKinsey & Company.* New York.

Barton, M., I. Ullah, S. Bergin, H. S. Sarjoughian, G. Mayer, J. Bernabeu-Auban, A. Heimsath, M. Acevedo, J. Riel-Salvatore, and R. Arrowsmith. 2016. "Experimental Socioecology: Integrative Science for Anthropocene Landscape Dynamics." *Anthropocene* 13: 34-45.

Daher, B. T., and R. H. Mohtar. 2015. "Water–Energy–Food (WEF) Nexus Tool 2.0: Guiding Integrative Resource Planning and Decision-Making." *Water International 40* 5-6: 748-771.

Dai, J., S. Wu, G. Han, W. Josh, X. Xie, X. Wu, X. Song, B. Jia, W. Xue, and Q. Yang. 2018. "Water-Energy Nexus: A Review of Methods and Tools for Macro-Assessment." *Applied energy 210* 393-408.

Fard, M. D., and H. S. Sarjoughian. 2019. "A Web-Service Framework for The Water Evaluation and Planning System." *Spring Simulation Conference (SpringSim)* 1-12.

Hamiche, A., A. Boudghene Stambouli, and S. Flazi. 2016. "A Review of The Water-Energy Nexus." *Renewable and Sustainable Energy* 65: 319-331.

Howells, M., S. Hermann, M. Welsch, M. Bazilian, R. Segerström, and et al. 2013. "Integrated Analysis of Climate Change, Land-Use, Energy and Water Strategies." *Nature Climate Change* 3: 621-626.

Huang, D., H. S. Sarjoughian, W. Wang, G. Godding, D. E. Rivera, K. G. Kempf, and H. Mittelmann. 2009. "Simulation of Semiconductor Manufacturing Supply-Chain Systems with DEVS, MPC, and KIB." *IEEE Transactions on Semiconductor Manufacturing* 22: 164-174.

Kalin, M. 2013. *Java Web Services: Up and Running: A Quick, Practical, and Thorough Introduction.* O'Reilly Media, Inc.

Khan, Z., P. Linares, M. Rutten, S. Parkinson, N. Johnson, and J. García-González. 2018. "Spatial and Temporal Synchronization of Water and Energy Systems: Towards a Single Integrated Optimization Model for Long-Term Resource Planning." *Applied Energy* 210: 499-517.

Kraucunas, I., L. Clarke, J. Dirks, J. Hathaway, M. Hejazi, K. Hibbard, and M. Huang. 215. "Investigating The Nexus of Climate, Energy, Water, and Land at Decision-Relevant Scales: The Platform for Regional Integrated Modeling and Analysis (PRIMA)." *Climatic Change* 129: 573-588.

2020. *LEAP.* Jan 1. Accessed Jan 1, 2020. https://www.energycommunity.org.

Martinez-Hernandez, E., M. Leach, and A. Yang. 2017. "Understanding Water-Energy-Food and Ecosystem Interactions Using The Nexus Simulation Tool NexSym." *Applied Energy* 206: 1009-1021.

Sarjoughian, H. S. 2006. "Model Composability." *Winter Simulation Conference.* Monterey, CA. 149--158.

Sieber, J., Swartz C., and Huber-Lee A. H. 2005. "Water evaluation and planning system (WEAP): User guide." *Stockholm Environment Institute.* Accessed March 17, 2020.

Van der Hoeven, M. 2013. *World Energy Outlook 2012.* Tokyo: International Energy Agency.

2020. *WEAP.* Jan 1. Accessed Jan 1, 2020. http://www.weap21.org/.

Yates, D., J. Sieber, D. Purkey, and A. Huber-Lee. 2005. "WEAP21— A Demand-, Priority-, and Preference-Driven Water Planning Model: Part 1: Model Characteristics." *Water International 30* 4: 487-500.

AUTHOR BIOGRAPHIES

MOSTAFA D. FARD is a Ph.D. student in the Computer Science program in the School of Computing, Informatics, and Decision Systems Engineering (CIDSE) at Arizona State University, Tempe, AZ, USA. He can be contacted at smd.fard@asu.edu.

HESSAM S. SARJOUGHIAN is an Associate Professor of Computer Science and Computer Engineering in the School of Computing, Informatics, and Decision Systems Engineering (CIDSE) at Arizona State University, and the co-director of the Arizona Center for Integrative Modeling & Simulation (ACIMS), Tempe, AZ, USA. His research interests include model theory, poly-formalism modeling, collaborative modeling, simulation for complexity science, and M&S frameworks/tools. He can be contacted at sarjoughian@asu.edu.