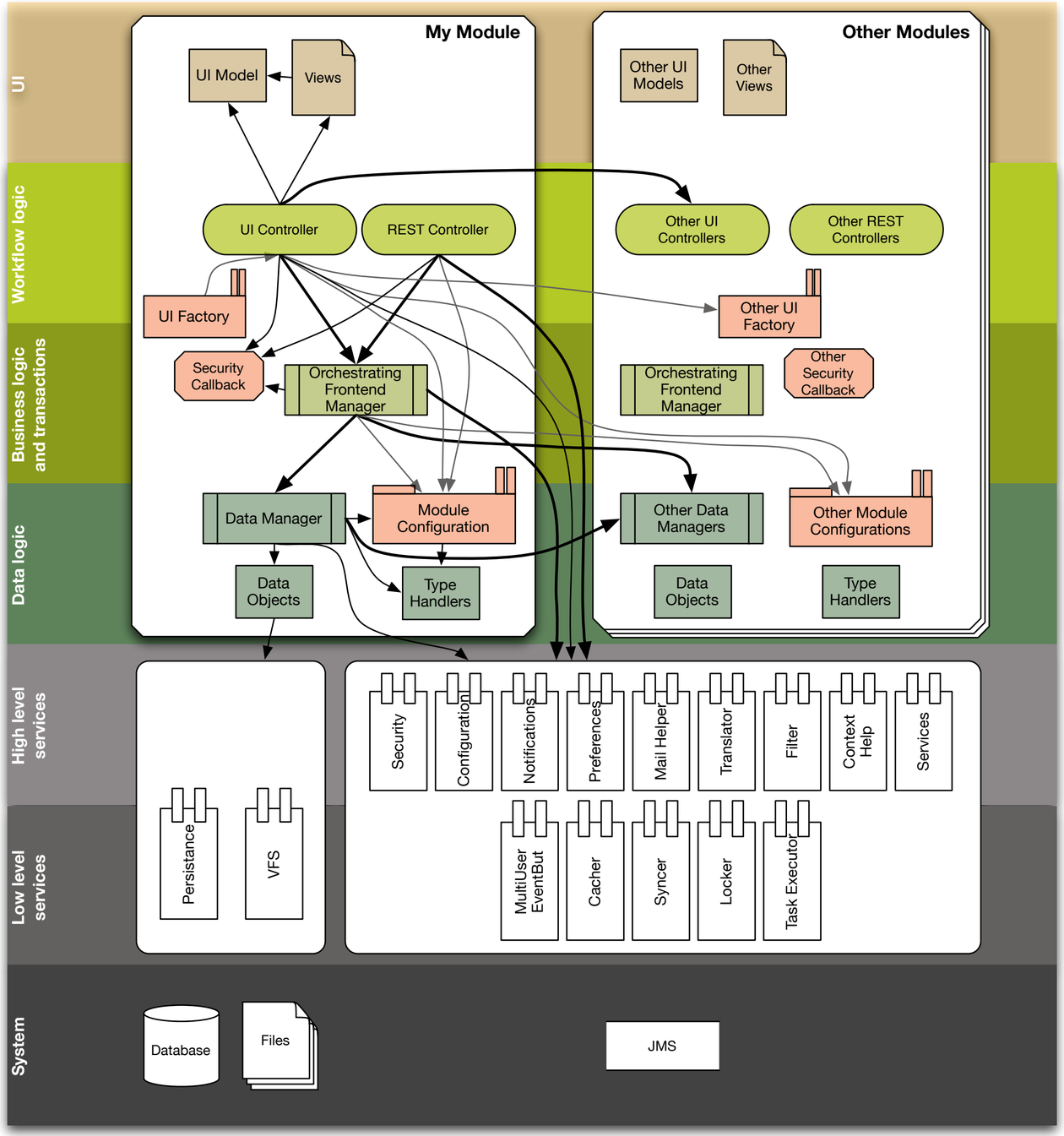
OLAT module and system architecture



Responsibilities

Module Configuration

- Of type AbstractOLATModule
- Spring registry/extension point and factory for module specific type handlers or other elements
- Registers module specific GUI elements in OLAT extension points
- Offers an admin GUI to enable/disable features or entire module
- Uses PersistedProperties to save admin configuration and propagate it in cluster

Type Handlers

- Spring beans that register in the module
- Implement type specific methods to deal with objects of that type
- Implement object and UI factories to display or manipulate objects of that type
- Optional, not used in all modules, but everywhere where a module deals with different objects of the same base type

Data Manager

- Implements typical CRUD operations on data objects with persistence
- Might use a framework cache to share loaded objects
- Provides factory method to create objects
- Provides methods to check validity of data fields (length checks, syntax etc)
- Methods do not check for security, trust contract to controller who is responsible for this
- No transactional context on changed data (does not manually execute commit)
- No synchronized context on changed data (does not call doInSync, no synchronized)
- Can use other data managers for sub operations
- Optimized database queries for finding and loading objects with complex joins or batch fetch strategies
- Responsible to do proper logging where not already done by data manager

Data Objects

- Dumb data object
- Provides only basic getter and setter methods for data fields
- No methods to manipulate object (e.g no persist methods)
- Data objects are manipulated only by the data manager

Orchestrating Frontend Manager

- Implements business rules: dependencies between objects, data integrity checks etc
- Methods do not check for security, trust contract to controller who is responsible for this.
- Provides methods to execute workflow steps
- Delegates database queries and other subtasks to data manager
- Can use other packages data manager for subtasks in a workflow step (Orchestrating)
- Transactional context, can open, execute and rollback transaction
- Synchronized context, can call doInSync
- Responsible to do proper logging where not already done by data manager

Security Callback

- Module specific security callback that offers feature check methods
- Normally created with a factory method by the frontend manager based on user permissions or dynamically based on users role in the current application context (derived permission)

UI Factory

- Provides factory methods to create and configure UI controllers. Can also be understood as UI Service

UI Controller

- Implements interface workflow logic
- Assembles UI using containers, components, child controllers, mappers and translators
- Responsible to cleanup child controllers and mappers during disposing
- Checks for security via security callback
- Delegates execution of workflow steps to frontend manager
- No direct doInSync or database calls
- Must implement context entry / business path
- Should implement activation via context entry / business path

Views / **UI Model**

- Arrangement of components in velocity containers with simple HTML markup
- Only display logic, no manager calls or other heavy logic
- Data model is preloaded by controller. No database calls or business logic execution during render phase

REST Controller

- Implements REST API logic
- Each method is stateless and independent
- Might implement caches to optimize typical set of subsequent stateless calls
- Checks for security via security callback
- Delegates execution to frontend manager
- No direct doInSync or database calls