METHOD AND SYSTEM FOR MULTI-LEVEL AUTOMATED STATUS MANAGEMENT IN CLINICAL TRIAL WORKFLOWS

CROSS-REFERENCE TO RELATED APPLICATIONS

BACKGROUND OF THE INVENTION

Field of the Invention

The present invention relates to computer-implemented systems for managing complex multi-entity clinical trial workflows, and more particularly to automated status computation and cross-entity validation systems for clinical research operations.

Description of Related Art

Clinical trials involve complex interactions between multiple entities including studies, protocol versions, sites, and participants, each with independent lifecycles but interdependent business rules. Traditional clinical trial management systems manage these entities in isolation, leading to significant operational and compliance challenges.

Existing systems suffer from several critical deficiencies. First, status inconsistencies arise because manual status updates create temporal gaps and are prone to human error. For example, when a protocol version is amended, manual processes may fail to properly update dependent study and site statuses, leading to operational confusion and potential regulatory violations.

Second, compliance violations occur when invalid state transitions violate FDA and International Council for Harmonisation (ICH) guidelines. Current systems lack integrated validation mechanisms to prevent transitions that would place a clinical trial in non-compliant states.

Third, operational inefficiencies result from the lack of automated workflow orchestration. Manual coordination between studies, sites, and protocol versions requires significant administrative overhead and increases the likelihood of errors.

Fourth, regulatory risks emerge from the inability to track and validate complex cross-entity dependencies. Regulatory agencies require comprehensive audit trails and validation that current systems cannot adequately provide.

Prior art systems such as Medidata Rave, Veeva Vault CTMS, Oracle Clinical, and IBM Clinical provide basic status management capabilities but fail to address the complex interdependencies inherent in multi-entity clinical trial workflows. These systems typically implement single-level status management without automated cross-entity validation or intelligent workflow orchestration.

Therefore, there exists a need in the art for a computer-implemented system that can automatically manage and coordinate status across multiple clinical trial entities while ensuring regulatory compliance and operational efficiency.

BRIEF SUMMARY OF THE INVENTION

The present invention provides a computer-implemented system for automated status management in clinical trial workflows that addresses the deficiencies of prior art systems through innovative multi-level status computation and cross-entity validation.

In accordance with one aspect of the present invention, a computer-implemented method for automated status management in clinical trial systems comprises: maintaining a plurality of clinical trial entities in a database, each entity having an independent status lifecycle; establishing dependency relationships between entities based on clinical trial business rules; automatically computing entity status based on states of dependent entities using a hierarchical status computation algorithm; validating status transitions across entity boundaries using cross-entity validation rules; and synchronizing status changes across dependent entities in real-time using event-driven propagation.

The hierarchical status computation algorithm gathers current states of all dependent entities, applies priority-based computation rules specific to clinical trial workflows, and determines optimal status considering regulatory compliance requirements.

Cross-entity validation includes constructing a real-time dependency graph of related entities, validating proposed status transitions against dependency constraints, and preventing transitions that would violate regulatory compliance rules.

The system provides significant advantages over prior art including reduced system complexity through automated status management, enhanced data integrity through cross-entity

validation, improved performance through event-driven architecture, and improved scalability through hierarchical design supporting large-scale multi-site trials.

BRIEF DESCRIPTION OF THE DRAWINGS

FIG. 1 is a block diagram illustrating the overall system architecture for multi-level automated status management.

FIG. 2 is a flowchart depicting the hierarchical status computation algorithm.

FIG. 3 is a diagram showing the cross-entity validation framework architecture.

FIG. 4 is a flowchart illustrating the event-driven state synchronization process.

FIG. 5 is a diagram showing dependency relationships between clinical trial entities.

DETAILED DESCRIPTION OF THE INVENTION

System Architecture Overview

Referring to FIG. 1, the multi-level automated status management system 100 comprises several key components working in coordination to manage clinical trial entity status. The system includes a hierarchical status computation engine 110, a cross-entity validation framework 120, an event-driven state synchronization component 130, and a compliance rule engine 140.

The system operates on clinical trial entities stored in a database 150, including but not limited to study entities 151, protocol version entities 152, site entities 153, and participant entities 154. Each entity maintains its own status lifecycle while participating in complex interdependency relationships managed by the system.

Hierarchical Status Computation Engine

The hierarchical status computation engine 110 implements a novel algorithm for determining entity status based on the states of dependent entities. The algorithm operates in multiple phases to ensure accurate and consistent status determination.

Referring to FIG. 2, the hierarchical status computation process 200 begins with gathering dependent entity states 210. The system queries the database to retrieve current status information for all entities that have dependency relationships with the target entity.

The process continues with applying hierarchical computation rules 220. These rules are specific to clinical trial workflows and incorporate regulatory compliance requirements. For example, a study entity cannot transition to ACTIVE status unless it has at least one approved protocol version and at least one active site.

The computation process includes priority-based status determination 230, where conflicting status indicators are resolved using predefined priority rules. For instance, if a study has multiple protocol versions in different states, the system applies priority rules to determine the overall study status.

Finally, the process outputs the computed status 240, which is then used to update the entity's status in the database and trigger any necessary downstream actions.

Cross-Entity Validation Framework

Referring to FIG. 3, the cross-entity validation framework 120 provides real-time validation of status transitions across entity boundaries. The framework constructs a dependency graph 310 that represents the current state and relationships of all relevant entities.

The validation process includes constraint checking 320, where proposed status transitions are validated against dependency constraints. For example, if a protocol version attempts to transition to SUPERSEDED status, the system validates that a newer version exists and is in an appropriate state.

Regulatory compliance validation 330 ensures that all status transitions comply with applicable regulations such as FDA 21 CFR Part 11 and ICH Good Clinical Practice guidelines.

The framework outputs validation results 340 that either approve the transition or provide detailed error information explaining why the transition is not permitted.

Event-Driven State Synchronization

Referring to FIG. 4, the event-driven state synchronization component 130 manages the propagation of status changes across dependent entities. When an entity status change occurs 410, the system identifies affected dependent entities 420.

The synchronization process determines required updates 430 for each affected entity based on the dependency rules and the nature of the triggering status change. For example, when a protocol version transitions to ACTIVE status, all associated sites may need to update their protocol deployment status.

The system executes synchronized updates 440 across all affected entities, ensuring that the updates occur in the correct order to maintain system consistency. The process includes rollback capabilities 450 in case of errors during the update process.

Compliance Rule Engine

The compliance rule engine 140 embeds regulatory business rules directly into the system logic, preventing invalid transitions that could result in regulatory violations. The rule engine includes predefined rule sets for major regulatory frameworks including FDA regulations, ICH guidelines, and other international standards.

The rule engine operates continuously, evaluating all proposed status transitions against applicable regulatory requirements. Rules are configurable and can be updated to accommodate changes in regulatory requirements without requiring system code changes.

Implementation Example

The following pseudocode illustrates the implementation of the hierarchical status computation algorithm:

```
BEGIN HierarchicalStatusComputation(entityId, entityType)
    dependentEntities = GatherDependentEntities(entityId, entityType)

    FOR EACH entity IN dependentEntities
        entityStates.add(entity.getCurrentStatus())
```

```
    END FOR

    computationContext = CreateComputationContext(entityStates)

    IF entityType == STUDY THEN
        computedStatus = ComputeStudyStatus(computationContext)
    ELSIF entityType == PROTOCOL_VERSION THEN
        computedStatus = ComputeProtocolVersionStatus(computationContext)
    ELSIF entityType == SITE THEN
        computedStatus = ComputeSiteStatus(computationContext)
    END IF

    validationResult = ValidateComputedStatus(entityId, computedStatus)

    IF validationResult.isValid THEN
        RETURN computedStatus
    ELSE
        THROW ValidationException(validationResult.getErrorMessage())
    END IF
END
```

The ComputeStudyStatus function implements priority-based logic:

```
BEGIN ComputeStudyStatus(computationContext)
    protocolVersions = computationContext.getProtocolVersions()
    studySites = computationContext.getStudySites()
    regulatoryStatus = computationContext.getRegulatoryStatus()

    IF HasActiveProtocolVersion(protocolVersions) AND
      AllSitesOperational(studySites) AND
      RegulatoryApprovalValid(regulatoryStatus) THEN
       RETURN ACTIVE
    END IF

    IF HasApprovedProtocolVersion(protocolVersions) AND
      SitesReadyForActivation(studySites) THEN
       RETURN APPROVED
    END IF

    RETURN ComputeTransitionalStatus(computationContext)
END
```

Cross-Entity Validation Implementation

The cross-entity validation framework implements the following algorithm:

```
BEGIN ValidateStatusTransition(entityType, entityId, fromStatus, toStatus)
    dependencyGraph = BuildDependencyGraph(entityType, entityId)

    FOR EACH dependency IN dependencyGraph.getDependencies()
        validationResult = ValidateDependencyConstraints(dependency, fromStatus, toStatus)
        IF NOT validationResult.isValid THEN
            RETURN validationResult
        END IF
    END FOR

    regulatoryValidation = ValidateRegulatoryCompliance(entityType, fromStatus, toStatus)
    RETURN regulatoryValidation
END
```

The BuildDependencyGraph function creates a real-time representation of entity relationships:

```
BEGIN BuildDependencyGraph(entityType, entityId)
    dependencyGraphBuilder = CreateDependencyGraphBuilder()

    SWITCH entityType
        CASE STUDY:
            RETURN dependencyGraphBuilder
```

```
            .addProtocolVersionDependencies(entityId)
            .addSiteDependencies(entityId)
            .addRegulatoryDependencies(entityId)
            .build()

        CASE PROTOCOL_VERSION:
            studyId = GetStudyIdForProtocol(entityId)
            RETURN dependencyGraphBuilder
                .addStudyDependency(studyId)
                .addAmendmentDependencies(entityId)
                .build()

        CASE SITE:
            RETURN dependencyGraphBuilder
                .addStudyDependencies(GetStudiesForSite(entityId))
                .addUserDependencies(entityId)
                .build()
    END SWITCH
END
```

Technical Advantages

The present invention provides several technical advantages over prior art systems. First, the automated status management reduces manual coordination overhead, decreasing the likelihood of human error and improving operational efficiency.

Second, cross-entity validation prevents inconsistent states that could lead to regulatory violations or operational confusion. The real-time validation ensures that the system maintains consistency at all times.

Third, the event-driven architecture enables real-time synchronization across multiple entities, ensuring that status changes are propagated immediately to all affected components.

Fourth, the hierarchical design supports scalability for large-scale multi-site clinical trials, accommodating complex dependency relationships without performance degradation.

Alternative Embodiments

While the present invention has been described with reference to clinical trial management, the multi-level automated status management system could be applied to other domains requiring complex entity relationship management, such as supply chain management, project management, or regulatory compliance systems.

The hierarchical status computation algorithm could be extended to support additional entity types or modified to accommodate different dependency relationship models.

The cross-entity validation framework could be enhanced with machine learning capabilities to predict potential validation issues before they occur.

The event-driven synchronization system could be implemented using different messaging architectures, such as publish-subscribe systems or message queues, depending on scalability and performance requirements.

CLAIMS

1. A computer-implemented method for automated status management in clinical trial systems, comprising:

Maintaining a plurality of clinical trial entities in a database, each entity having an independent status lifecycle;

Establishing dependency relationships between entities based on clinical trial business rules;

Automatically computing entity status based on states of dependent entities using a hierarchical status computation algorithm;

Validating status transitions across entity boundaries using cross-entity validation rules; and

Synchronizing status changes across dependent entities in real-time using event-driven propagation.

2. The method of claim 1, wherein the hierarchical status computation algorithm comprises:

Gathering current states of all dependent entities;

Applying priority-based computation rules specific to clinical trial workflows; and

Determining optimal status considering regulatory compliance requirements.

3. The method of claim 1, wherein cross-entity validation comprises:

Constructing a real-time dependency graph of related entities;

Validating proposed status transitions against dependency constraints; and

Preventing transitions that would violate regulatory compliance rules.

4. The method of claim 1, wherein the clinical trial entities include at least study entities, protocol version entities, site entities, and participant entities.

5. The method of claim 1, wherein the dependency relationships are based on regulatory requirements including FDA 21 CFR Part 11 and ICH Good Clinical Practice guidelines.

6. The method of claim 1, wherein the event-driven propagation includes rollback capabilities for handling errors during synchronization.

7. The method of claim 2, wherein the priority-based computation rules resolve conflicting status indicators using predefined priority hierarchies.

8. The method of claim 3, wherein the dependency graph is constructed dynamically based on current entity states and relationships.

9. A computer system for automated status management in clinical trial workflows, comprising:

A processor;

A memory coupled to the processor;

A database storing a plurality of clinical trial entities, each entity having an independent status lifecycle;

A hierarchical status computation engine configured to automatically compute entity status based on states of dependent entities;

A cross-entity validation framework configured to validate status transitions across entity boundaries; and

An event-driven state synchronization component configured to synchronize status changes across dependent entities in real-time.

10. The computer system of claim 9, further comprising a compliance rule engine configured to embed regulatory business rules and prevent invalid transitions.

11. The computer system of claim 9, wherein the hierarchical status computation engine implements priority-based status determination for resolving conflicting status indicators.

12. The computer system of claim 9, wherein the cross-entity validation framework constructs real-time dependency graphs for validation.

13. A non-transitory computer-readable storage medium storing instructions that, when executed by a processor, cause the processor to:

Maintain a plurality of clinical trial entities in a database, each entity having an independent status lifecycle;

Establish dependency relationships between entities based on clinical trial business rules;

automatically compute entity status based on states of dependent entities using a hierarchical status computation algorithm;

Validate status transitions across entity boundaries using cross-entity validation rules; and

Synchronize status changes across dependent entities in real-time using event-driven propagation.

14. The non-transitory computer-readable storage medium of claim 13, wherein the instructions further cause the processor to implement regulatory compliance validation according to FDA and ICH guidelines.

15. The non-transitory computer-readable storage medium of claim 13, wherein the instructions further cause the processor to provide rollback capabilities for handling synchronization errors.

ABSTRACT

A computer-implemented system for automated status management in clinical trial workflows provides multi-level status computation and cross-entity validation. The system maintains clinical trial entities including studies, protocol versions, sites, and participants, each with independent status lifecycles but interdependent business rules. A hierarchical status computation engine automatically computes entity status based on dependent entity states using

priority-based computation rules and regulatory compliance requirements. A cross-entity validation framework validates status transitions by constructing real-time dependency graphs and preventing transitions that would violate regulatory compliance rules. Event-driven state synchronization propagates status changes across dependent entities in real-time with rollback capabilities. The system provides enhanced data integrity, regulatory compliance, and operational efficiency compared to prior art systems that manage entities in isolation.