# Job History Model

The life history of an SPK job is represented as a finite state automaton.

## Table of Contents



## Introduction

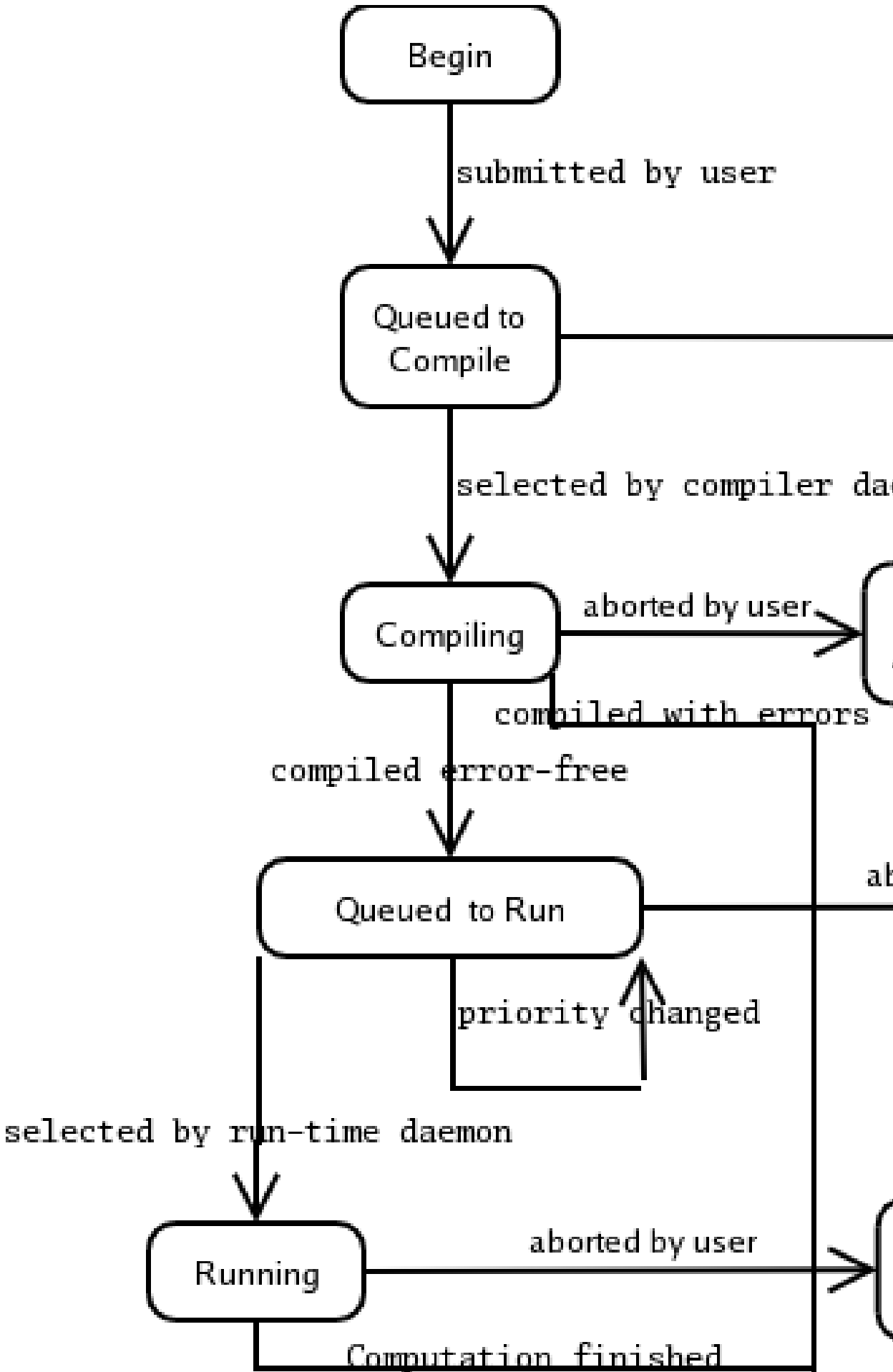
A job is created when the user combines a model with data and submits this package to be run by Spk. Submitting jobs is one of the functions of the Model Design Agent (MDA), the graphical interface that runs on the user's workstation. The job is sent via the Internet to the Application Server (ASPK) to be translated into computer source code. Once compiled, it is sent onward, via the Internet to the Computational Server (CSPK) to be translated into machine code and executed.

The life cycle of a job can be represented as a sequence of states. When a user inquires about the status of one of her outstanding jobs, the answer is based on its current state. The history of a job is the sequences of states it has assumed, along with the times of transition from one state to another and any outputs that have occurred. The purpose of this document is to define the set of states that a job may pass through, and the rules governing transitions from one state to the next.

## Job State Diagram

The following diagram represents the life cycle of a job as a finite state automaton. Four kinds of elements are present:

- States are represented by boxes.
- Transitions are represented by arrows.
- An event is represented by text appearing near the beginning of the arrow representing the transition caused by the event.
- An output is represented by text appearing near the end of a transition arrow. Not every transition produces output. To further distinguish an output from an event, the name of the output is preceded by the '>' character.


Begin
submitted by user
Queued to Compile
selected by compiler da
Compiling
aborted by user
compiled with errors
compiled error-free
Queued to Run
priority changed
selected by run-time daemon
Running
aborted by user
Computation finished

## Description of Elements

### Begin

This state is purely formal. The job is in this state until it is submitted.

### submitted by MDA

This event occurs when the user commands the MDA to submit the job. At that time, the job makes the transition to the *Queued to Compile* state.

### Queued to compile

The job is waiting in a prioritized queue until sufficient resources become available for the compiler to translate it from model specifications to computer source code.

### selected by ASPK compiler

occurs when compilation resources are available, there is no job of higher priority and there is no job of equal priority that has been waiting longer.

### aborted by user

occurs when the user has requested to abort the job through the MDA.

### aborted

When aborted the job comes to its *End* state.

### Compiling

The compiler is processing the model specification.

### compiled error-free

When this event occurs, the job is ready to be queued for running

#### compiled with errors

If there are errors in the compilation, the job cannot be queued to run. Instead, the next state is *End*.

An error report is output on this transition.

#### aborted by user

occurs when the user has requested to abort the job through the MDA. The next state is *Queued to abort compiling*.

#### aborted

When aborted the job comes to its *End* state.

#### Queued to run

The job is waiting in a prioritized queue for resources to assigned to it so that it can run.

#### selected to run

The job is selected when sufficient resources become available for it to run, there is no job of higher priority in the queue, and there is no job of equal priority that has been waiting longer.

#### priority changed

The priority of the job is changed. It makes the transition back to the same state, but with the new priority.

#### aborted by user

occurs when the user has requested to abort the job through the MDA.

#### aborted

When aborted the job comes to its *End* state.

### Running

The Computational Server (CSPK) is executing the job.

### finished

When finished (whether the problem converges satisfactorily or not) the job comes to its *End* state.

A report is output on this final transition.

### aborted by user

occurs when the user has requested to abort the job through the MDA. The next state is *Queued to abort computation*.

### Queued to Abort Compiling

The job is waiting in a prioritized queue until sufficient resources become available for the compiler daemon to end the compilation.

### selected by compiler daemon

occurs when compiler daemon is available, there is no aborting job of higher priority and there is no aborting job of equal priority that has been waiting longer.

### Aborting Compilation

The compiler daemon is aborting the job.

### aborted

When aborted the job comes to its *End* state.

### Queued to Abort Running

The job is waiting in a prioritized queue until sufficient resources become available for the run-time daemon to end the computation.

### selected by run-time daemon

occurs when run-time daemon is available, there is no aborting job of higher priority and there is no aborting job of equal priority that has been waiting longer.

### Aborting Computation

The run-time daemon is aborting the job.

### aborted

When aborted the job comes to its *End* state.

### End

This is the final state of every job.