Solr Improvement Proposals

This page describes a proposed Solr Improvement Proposal (SIP) process for proposing a major change to Solr.

To create your own SIP, click on "Create" on the header and choose "SIP-Template", or click here.

- Purpose
- What is considered a "major change" that needs a SIP?
- What should be included in a SIP?
- Who should initiate the SIP?
- Process
- SIP round-up
- SIPs under discussion
- Adopted/Accepted but unreleased SIPs
- Implemented and Released SIPs
- Discarded SIPs

Purpose

The purpose of SIPs is to have a central place to collect and document planned major enhancements to Apache Solr. While JIRA is still the tool to track tasks, bugs, and progress, the SIPs give an accessible high level overview of the result of design discussions and proposals. Think of SIPs as collections of major design documents for user-relevant changes.

We want to make Solr well architected with consistent public APIs. We will also support plugins/packages, which requires a commitment to a high level of compatibility between releases — core architectural elements can't break compatibility or shift functionality from release to release. As a result each new major feature or public API has to be done in a way that we can stick with it going forward.

This means when making this kind of change we need to think through what we are doing as best we can prior to release. And as we go forward we need to stick to our decisions as much as possible. All technical decisions have pros and cons so it is important we capture the thought process that lead to a decision or design to avoid SIP-flopping needlessly.

Hopefully we can make these proportional in effort to their magnitude — small changes should just need a couple brief paragraphs, whereas large changes need detailed design discussions.

This process also isn't meant to discourage incompatible changes — proposing an incompatible change is totally legitimate. Sometimes we will have made a mistake and the best path forward is a clean break that cleans things up and gives us a good foundation going forward. Rather this is intended to avoid accidentally introducing half thought-out interfaces and protocols that cause needless heartburn when changed. Likewise the definition of "compatible" is itself squishy: small details like which errors are thrown when are clearly part of the contract but may need to change in some circumstances, likewise performance isn't part of the public contract but dramatic changes may break use cases. So we just need to use good judgement about how big the impact of an incompatibility will be and how big the payoff is.

What is considered a "major change" that needs a SIP?

Any of the following should be considered a major change:

- Any major new feature, subsystem, or piece of functionality
- Any change that impacts the public interfaces of the project

What are the "public interfaces" of the project?

All of the following are public interfaces that people build around:

- All interfaces published for plugin authors (QParser, UpdateProcessor, ResponseWriter, SearchComponent etc)
- On-disk binary formats, such as index and backup structure
- User-facing scripts/command-line tools, i.e. bin/solr
- Layout of Zookeeper (i.e. location of configsets)
- Format of important ZK nodes
- Configuration files (such as schema.xml, solrconfig.xml, security.json, clusterprops.json)
- Exposed monitoring information

Not all compatibility commitments are the same. We need to spend significantly more time on public APIs as these can break code for users. They cause people to rebuild code and lead to compatibility issues in large multi-dependency projects (which end up requiring multiple incompatible versions). Configuration, monitoring, and command line tools can be faster and looser — changes here will break monitoring dashboards and require a bit of care during upgrades but aren't a huge burden.

For the most part monitoring, command line tool changes, and configs are added with new features so these can be done with a single SIP.
What should be included in a SIP?

A SIP should contain the following sections:

- **Motivation**: describe the problem to be solved
- **Proposed Change**: describe the new thing you want to do. This may be fairly extensive and have large subsections of its own. Or it may be a few sentences, depending on the scope of the change.
- **New or Changed Public Interfaces**: impact to any of the “compatibility commitments” described above. We want to call these out in particular so everyone thinks about them.
- **Migration Plan and Compatibility**: if this feature requires additional support for a no-downtime upgrade describe how that will work
- **Performance impact**: How will the change affect performance? Benchmarking
- **Rejected Alternatives**: What are the other alternatives you considered and why are they worse? The goal of this section is to help people understand why this is the best solution now, and also to prevent churn in the future when old alternatives are reconsidered.

Who should initiate the SIP?

Anyone can initiate a SIP but you shouldn't do it unless you have an intention of getting the work done to implement it (otherwise it is silly).

Process

Here is the process for making a SIP:

1. If you don't have the necessary permissions for creating a new page, please ask on the development mailing list.
2. Create a page which is a child of this one. Take the next available SIP number and give your proposal a descriptive heading. e.g. “SIP 42: Solr API v3”.
3. Fill in the sections as described above
4. Start a [DISCUSS] thread on the mailing list. Please ensure that the subject of the thread is of the format [DISCUSS] SIP-{your SIP number} {your SIP heading}. Discussion can also happen on the WIKI page for the SIP by using Confluence inline commenting system (Mark the words you want to comment and click the comment icon that pops up).
5. When feedback iteration trails off on the DISCUSS thread, move to either adopt the proposal. To do this you can either:
   a. rely on lazy consensus - announce your intention to proceed with the SIP as written unless there are any further objections, then wait 3 business days for readers to object or request an explicit VOTE instead.
   b. Alternatively, you can jump right to calling a formal VOTE. Create a VOTE thread that points readers to your proposal and ask for votes.
      The criteria for acceptance is consensus (3 binding +1 votes and no binding vetoes). Once a VOTE thread is created, add a link to the table below so the outcome can be tracked clearly.
6. If the proposal was accepted, move the SIP from the "Under Discussion" table to the "Adopted" table below. If the proposal failed to pass, move the SIP to the "Discarded" table.
7. Implement the SIP.
8. When implementation is complete, move the SIP to the "Implemented" table below.

SIP round-up

**Next SIP Number: 16**

Use this number as the identifier for your SIP and increment this value.

SIPs under discussion

<table>
<thead>
<tr>
<th>SIP</th>
<th>State</th>
<th>Link to JIRA</th>
<th>Link to dev@Discussion Thread</th>
</tr>
</thead>
<tbody>
<tr>
<td>SIP-1: Streamline Config/Schema APIs</td>
<td>Discuss</td>
<td>SOLR-14041 - Getting issue details...</td>
<td>[STATUS]</td>
</tr>
<tr>
<td>SIP-2 Support safe index transformations without reindexing</td>
<td>Discuss</td>
<td>SOLR-14116 - Getting issue details...</td>
<td>[STATUS]</td>
</tr>
<tr>
<td>SIP-3 Solr-specific log and thread indexing and UI</td>
<td>Discuss</td>
<td>SOLR-14121 - Getting issue details...</td>
<td>[STATUS]</td>
</tr>
<tr>
<td>SIP-4 Resource management framework</td>
<td>Discuss</td>
<td>SOLR-13579 - Getting issue details...</td>
<td>[STATUS]</td>
</tr>
<tr>
<td>SIP-5 Coordination Module + Apache Curator</td>
<td>Discuss</td>
<td>SOLR-13578 - Getting issue details...</td>
<td>[STATUS]</td>
</tr>
<tr>
<td>SIP-6 Solr should own the bootstrap process</td>
<td>Discuss</td>
<td>SOLR-14361 - Getting issue details...</td>
<td>Mail thread</td>
</tr>
<tr>
<td>SIP</td>
<td>Target Release</td>
<td>Link to JIRA</td>
<td>Link to dev@ Discussion Thread</td>
</tr>
<tr>
<td>----------------------------</td>
<td>----------------</td>
<td>------------------------------</td>
<td>--------------------------------</td>
</tr>
<tr>
<td>SIP-7 Updated Solr Admin UI</td>
<td>Discuss</td>
<td>SOLR-14414 - Getting issue details...</td>
<td>Mail thread</td>
</tr>
<tr>
<td>SIP-9 Advanced Query Parser</td>
<td>Discuss</td>
<td>SOLR-14597 - Getting issue details...</td>
<td>Mail thread</td>
</tr>
<tr>
<td>SIP-11 Uniform cluster-level configuration API</td>
<td>Discuss</td>
<td>SOLR-14843 - Getting issue details...</td>
<td>Slack thread</td>
</tr>
<tr>
<td>SIP-13: Cross Data Center Replication</td>
<td>Discuss</td>
<td>SOLR-15636 - Getting issue details...</td>
<td>Mail thread</td>
</tr>
<tr>
<td>SIP-14 Embedded Zookeeper</td>
<td>Discuss</td>
<td>SOLR-15694 - Getting issue details...</td>
<td>Mail thread</td>
</tr>
<tr>
<td>SIP-15 Node roles</td>
<td>Discuss</td>
<td>SOLR-15694 - Getting issue details...</td>
<td>Mail thread</td>
</tr>
</tbody>
</table>

**Adopted/Accepted but unreleased SIPs**

<table>
<thead>
<tr>
<th>SIP</th>
<th>Target Release</th>
<th>Link to JIRA</th>
<th>Link to dev@ Discussion Thread</th>
</tr>
</thead>
<tbody>
<tr>
<td>SIP-8 Autoscaling policy engine V2</td>
<td>Discuss</td>
<td>SOLR-14275 - Getting issue details...</td>
<td>Slack thread</td>
</tr>
</tbody>
</table>

**Implemented and Released SIPs**

<table>
<thead>
<tr>
<th>SIP</th>
<th>First Release Version</th>
<th>Link to JIRA</th>
<th>Link to dev@ Discussion Thread</th>
</tr>
</thead>
<tbody>
<tr>
<td>SIP-12 Incremental Backup and Restore</td>
<td>8.9</td>
<td>SOLR-15086 - Getting issue details...</td>
<td>Mail thread</td>
</tr>
</tbody>
</table>

**Discarded SIPs**

<table>
<thead>
<tr>
<th>SIP</th>
<th>Comment</th>
</tr>
</thead>
</table>