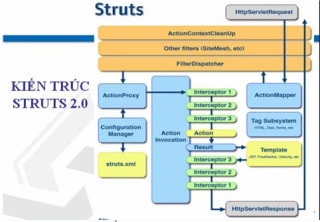
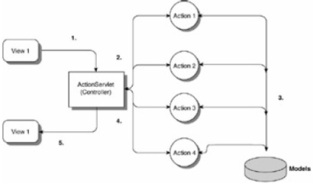
**Struts Framework(Struts 2.0)**

[](http://1.bp.blogspot.com/-h4A4ZyUxA4Y/TtpLJVtl-lI/AAAAAAAAAYA/MxPOQ8VkOZQ/s1600/struts09.jpg)

Kiến trúc của Struts 2.0

**1.    Khái niệm:**  
Struts là một framework phục vụ việc phát triển các ứng dụng Web trên Java. Struts cung cấp một framework thống nhất để deploy(triển khai) các ứng dụng Servlet và JSP sử dụng kiến trúc MVC. Sử dụng mẫu thiết kế Model-View-Controller (MVC), Struts giải quyết rất nhiều các vấn đề liên quan đến các ứng dụng Web hướng business đòi hỏi hiệu năng cao sử dụng Java servlet và JSP. Struts cơ bản định hình lại cách các Web programmer nghĩ về và cấu trúc một ứng dụng Web.  
Struts là một tập thư viện các thẻ JSP tùy chọn (Custom JSP Tag): Struts cung cấp các thư viện thẻ tùy chọn cho việc thể hiện các thuộc tính của bean, quản lý các HTML forms, lặp lại các kiểu cấu trúc dữ liệu, và đưa ra các HTML có điều kiện.

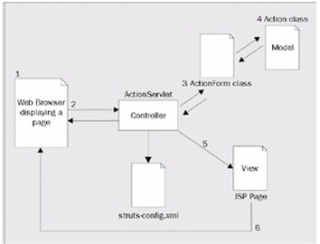
**2.    Tiến trình thực thi mô hình MVC của Struts như thế nào?**

[](http://2.bp.blogspot.com/-ABrLCRs5znA/TtpLKD_2MdI/AAAAAAAAAYE/NGWlP_Buusg/s1600/struts10.jpg)

Mô tả tiến trình mà hầu hết các ứng dụng struts phải tuân theo

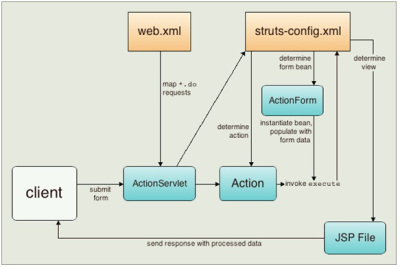
Tiến trình này có thể được chia thành 5 bước cơ bản sau:  
1. Một request được gửi đến từ view  
2. ActionServlet sẽ tiếp nhận request này, phân tích, kiểm tra. Sau đó chỉ định cho Action tương ứng thực thi yêu cầu, tính toán những tác vụ cần thiết.  
ActionServlet đóng vai trò là **Controller**  
3. Action sẽ thao tác và xử lí trên Model của ứng dụng  
4. Mỗi khi Action hoàn thành việc thao tác và xử lí, nó trả quyền điều khiển về cho ActionServlet kèm theo một key gắn kèm với kết quả trả về. ActionServlet sẽ dựa vào key này mà quyết định xem các kết quả trả về sẽ được hiển thị như thế nào.  
5. ActionServlet trả lời bằng cách gửi lại một request cho view là một liên kết đến kết qua trả về của Action thông qua key trên. Sau đó, view làm nốt công việc trình bày kết quả.  
Nguồn benalpenliebe.blogspot.com

**3.    Cấu trúc của Struts:**

[](http://4.bp.blogspot.com/-FWaxLw8z5YU/TtpLKsurMgI/AAAAAAAAAYQ/JwQbee7iu90/s1600/struts11.jpg)

Cấu trúc của Struts

•    Một hoặc nhiều Action, mỗi action trong trang web sẽ ánh xạ chính xác đến một thành phần <action> được định nghĩa trong file struts-config.xml. Action được triệu gọi bởi người dùng từ một trang HTML hoặc JSP thông qua một liên kết hay thông qua thuộc tính action trong thẻ <form>.  
•    Một thành phần <action> sẽ định nghĩa một lớp ActionForm, trong một số trường hợp, nó sẽ được sử dụng để validate(xác nhận) các dữ liệu trong form được submit bởi người sử dụng. Nó cũng định nghĩa lớp Action nào sẽ được sử dụng để xử lý các yêu cầu từ phía người dùng.  
•    Một lớp ActionForm có thể sử dụng một hoặc nhiều forward được định nghĩa trong thẻ <action> đề nói cho một ActionServlet trả về các response tương ứng với các request của người dùng. Chúng ta có thể định nghĩa nhiều forward trong thẻ <action-mapping>.  
**4.    Các thành phần chính của một ứng dụng struts:**  
**Struts Model Components**  
Model là một thành phần được cho là quan trọng nhất trong các ứng dụng MVC.  
Model bao gồm các business entities và một tập các qui tắc để quản lí việc tổ chức và thao tác dữ liệu.  
Struts không cung cấp các Model Component chuyên dụng, tuy nhiên chúng ta có thể sử dụng lại các Model của các ứng dụng khác hoặc tự xây dựng các model của riêng mình.  
**Struts View Components**  
Mục đích của thanh phần Struts View này cũng giống y như một thành phần view trong các ứng dụng theo mô hình MVC: chịu trách nhiệm trình bày thông tin được cung cấp bởi Model.  
Struts sử dụng JSP để thiết kế thành phần View. Ngoài ra, để hỗ trợ và mở rộng khả năng của View, chúng ta cũng có thêm thư viện Taglib, sử dụng HTML, JS… cho mục đích trình bày thông tin.  
**Struts Controller Components**  
Struts cung cấp hai thành phần rất quan trọng đó là ActionServlet và Action để điều khiển và quản lí mọi yêu cầu của người dùng cũng như việc thao tác với dữ liệu  
ActionServlet chịu trách nhiệm nhận và xử lí các request từ phía người dùng, chỉ định Action thực thi tương ứng với từng yêu cầu cụ thể.  
Action chịu trách nhiệm thao tác với Model, nó kết hợp rất chặt chẽ với ActionServlet. Cả hai thành phần này đóng vai trò làm Controller trong Struts.  
Struts taglib  
Cung cấp một tập các tag library cho việc phát triển ứng dụng, bao gồm cả các taglib hỗ trợ thiết kế HTML và JSP taglib.  
Struts Config  
File cấu hình của ứng dựng Struts.  
**5.    Cơ chế hoạt động:**

[](http://2.bp.blogspot.com/-MZ88W7G22no/TtpLL5PMqcI/AAAAAAAAAYU/zBGKSl0qDHU/s1600/struts12.jpg)

1. Ứng dụng web mà bạn phát triển có một mô tả triển khai (/web.xml) mà bạn phải viết. Tập tin này mô tả cấu hình ứng dụng web của bạn, bao gồm trang khởi động, ánh xạ các servlet (đuờng dẫn hoặc phần mở rộng), và các tham số truyền tới các servlet đó.  
Trong tập tin web.xml,bạn phải cấu hình ActionServlet làm servlet xử lý tất cả các yêu cầu đối với một ánh xạ cho truớc (thuờng sử dụng phần mở rộng .do). ActionServlet chính là “bàn điều khiển” đuợc nhắc tới trong phần mở đầu của bài viết. Cũng trong tập tin web.xml, cấu hình cho ActionServlet sử dụng một hoặc nhiều tập tin cấu hình cho bản thân Struts.  
2. Trong tập tin cấu hình framework, bạn liên kết các đuờng dẫn với thành phần điều khiển của ứng dụng, chính là các lớp Action (ví dụ như “login” => lớp LoginAction). Việc liên kết này thông báo cho ActionServlet rằng để đáp ứng yêu cầu gọi thành phần điều khiển của bạn, là LoginAction.  
Về phần mở rộng do trong URL. Phần mở rộng sẽ khiến máy chủ của bạn (Tomcat chẳng hạn) gọi ActionServlet. Cấu hình đuợc tham chiếu và LoginAction đuợc thực thi.  
3. Với mỗi Action,bạn cũng cấu hình framework với tên của các trang kết quả có thể đuợc hiển thị tuơng ứng với Action đó. Có thể có nhiều hiển thị đuợc dùng làm kết quả của một Action (thuờng có ít nhất là hai: một tuơng ứng với kết quả thành công và một ứng với kết quả thất bại).  
Action của bạn (phần điều khiển mà bạn viết) đuợc dựa trên các tên ánh xạ kết quả logic này. Nó thông báo cho ActionServlet bằng các từ như “success”, “failure”, “ready”, “ok”, “UserError”, vv… Framework (thông qua cấu hình mà bạn đã thiết lập) sẽ biết cách chuyển tới các trang thích hợp. Việc cấu hình này tận dụng ưu thế của việc cấu hình lại đối với lớp hiển thị bằng cách chỉnh sửa các tập tin cấu hình XML.  
Tới lúc này, framework đã biết cách ủy quyền cho các thành phần điều khiển của bạn và các kết quả cần đuợc hiển thị sau khi các thành phần điều khiển xử lí. Phần mô hình (model) của ứng dụng sẽ hoàn toàn phụ thuộc vào bạn, và sẽ đuợc gọi từ các thành phần điều khiển của bạn.  
4. Bạn cũng có thể liên kết một JavaBean với một action (hoặc một tập hợp nhiều action) trong tập tin cấu hình của framework. JavaBean đuợc sử dụng như một kho lưu trữ cho form hoặc hiển thị dữ liệu có thể đuợc kết nối giữa lớp hiển thị và lớp điều khiển.  
Các JavaBean này cũng có thể đuợc truy cập từ các thành phần điều khiển của bạn (như lớp LoginAction) và từ các trang hiển thị được liên kết với thành phần điều khiển đó. Chúng cũng có thể được kiểm tra với sự trợ giúp của framework để đảm bảo rằng người dùng nhập thông tin chính xác vào các form.  
Lưu ý rằng bạn phải sử dụng một công nghệ máy chủ nào đó (JSP, Velocity, XSLT) để lớp hiển thị nhìn thấy các dữ liệu này (dạng HTML thuần không dùng được). Framework hoạt động ở phía máy chủ nên phần hiển thị ở máy khách cần được tạo ra ở đó. Máy kháh đẩy dữ liệu trở lại thông qua phương thức thông thường của form (POST/GET), và framework sẽ cập nhật các dữ liệu đó trong Bean trước khi gọi các thành phần điều khiển.  
5. Trong ứng dụng web sẽ có các trang hiển thị mà người dùng nhìn thấy. Đó có thể là các trang JSP, các mẫu Velocity, các trang XSLT, … Tập các trang JSP và thẻ JSTL có sẵn trong framework cho bạn sử dụng nhưng bạn có thể sử dụng bất cứ công nghệ hiển thị chuẩn nào cũng được. Ngay cả các tập tin HTML thuần cũng có thể được dùng trong ứng dụng nhưng chúng sẽ khong tận dụng được các tính năng động.  
Theo sau thành công của thư viện thẻ JSP, một số gói khác đuợc tạo ra để framework đuợc sử dụng dễ dàng hơn với công nghệ hiển thị ưa thích của bạn. Với các mẫu Velocity, có công cụ hiển thị cho Velocity bạn. Nếu bạn muốn dùng XSLT trong ứng dụng, bạn có thể sử dụng stxx hoặc StrutsCX.  
Các gói này giúp các phần tử chuẩn của Struts đồng nhất với công nghệ hiển thị gốc. Bạn cũng có thể sử dụng cả JSP, Velocity và XSLT trong cùng một ứng dụng!  
Do Struts phụ thuộc vào công nghệ Servlet chuẩn, bạn có thẻ sử dụng bất cứ công nghệ hiển thị Java nào với nó.  
6. Mặc dù trọng tâm của Struts là ở thành phần điều khiển (C), lớp hiển thị là một thành phần quan trọng của bất kì ứng dụng nào. Thư viện thẻ Struts có một số thẻ đặc trưng dể giúp bạn sử dụng các dữ liệu động trong phần hiển thị của mình.  
Các thẻ JSP góp một phần đáng kể trong mã nguồn cơ sở của Struts. Trong phiên bản 1.1b3, khi mà mã Java cho phần lõi của Struts chỉ gồm khoảng 28000 dòng thì phần mã Java cho thư viện thẻ đã lên tới 41000 dòng.  
Các thẻ này giúp bạn kết nối lớp hiển thị (V) với lớp điều khiển (C) mà không phải nhúng quá nhiều mã Java trong trang JSP. Việc này giúp trang hiển thị có dạng như XML và dễ dàng hơn cho công việc của các nhà thiết kế web. Nó cũng giúp giảm thiểu các lệ thuộc giữa phần điều khiển và phần hiển thị.  
Có cả các thẻ gúp bạn quốc tế hoá, hiển thị thông báo lỗi, …  
Tất cả các tính năng này phụ thuộc vào các tập tin cấu hình mà bạn cung cấp cho Struts.  
Bạn cũng cần ghi nhớ rằng cơ chế đuợc mô tả ở đây chỉ có tác dụng khi các yêu cầu đuợc xử lí bởi ActionServlet.Và điều này chỉ xảy ra khi yêu cầu từ trình duyệt khiến máy chủ (như Tomcat, WebSphere, …) gọi ActionServlet. Do đó bạn cần đảm bảo rằng các trang phụ thuộc vào Struts đuợc gọi thông qua một yêu cầu đuợc ánh xạ tới ActionServlet (có phần mở rộng .do

**6.    Thư viện thẻ Struts để xây dựng các thành phần trình diễn trong một ứng dụng:**  
+ <templete>: cung cấp cho nhà phát triển ứng dụng một tập các thẻ JSP để chia nhỏ giao diện người dùng thành các thành phần có thể dễ dàng tháo rắp.  
+ <bean>: cung cấp cho nhà phát triển ứng dụng một tập các thẻ JSP để quản lý đầu ra từ một JavaBean.  
+ <logic>: có thể được sử dụng để ứng dụng các điều kiện logic trong một trang JSP.  
+ <html>: có thể sử dụng để tạo ra các thành phần form  
**7.    Ưu điểm của Struts (so với MVC sử dụng requestDispatcher):**  
•    Struts 2 được xây dựng và phát triển dựa trên nền tảng mô hình MVC nên nó thừa hưởng được đầy đủ các ưu điểm mà mô hình MVC đem lại.  
•    Dễ dàng tùy chỉnh (customize) chu kỳ xử lý (request lifecycles ) cho từng action  
•    Giải quyết hiệu quả vấn đề internationlization và localization trong các ứng dụng web  
•    Tự động chuyển đổi kiểu dữ liệu chuỗi truyền thống trong tham số request parameter thành các đối tượng lớp dữ liệu java => tiết kiệm được thời gian và công sức cho các lập trình viên  
•    Cung cấp các thẻ tag,các themes và templates giúp cho việc làm giao diện GUI trở nên dễ dàng,nhanh lẹ và tăng tính tái sử dụng.  
•    Tính mở rộng (Extensibility) cao thông qua việc hỗ trợ các plug-in  
•    Hỗ trợ portal.  
•    Hỗ trợ AJAX  
•    Dễ dàng tích hợp với Spring framework(\*) và Hibernate.  
**8.    Nhược điểm của Struts:**  
Để sử dụng MVC với chuẩn RequestDispatcher, ta cần nghiên cứu sâu với chuẩn JSP và Servlet APIs. Để sử dụng MVC với Struts, ta còn phải hiểu rõ cả framework rộng lớn và phức tạp, nó tương tự như việc tìm hiểu cả cái cốt lõi của hệ thống. Điều bất lợi này đặc biệt đáng kể với những dự án nhỏ, những dự án có ít thời gian để thực hiện, và những lập trình viên có ít kinh nghiệm; ta mất rất nhiều thời gian trong việc nghiên cứu Struts khi thực hiện đề án.   
Framework là một thư viện, bộ khung để phát triển các phần mềm ứng dụng. Tức là nó tạo ra các “Vật liệu” ở từng lĩnh vực cho người lập trình viên, thay vì họ phải mất nhiều thời gian để tự thiết kế trước khi dùng. Do vậy, người lập trình viên chỉ cần tìm hiểu và khai thác các vật liệu này rồi thực hiện (tức lập trình) để gắn kết chúng lại với nhau, tạo ra sản phẩm.  
Framework có 2 thành phần chính là Common Language Runtime (CLR) và NET Framework class library. CLR là môi trường được dùng để quản lý sự thi hành các nguồn mã mà ta đã soạn ra và biên dịch trong các ứng dụng. Tuy nhiên khi biên dịch nguồn mã, ta lại biên dịch chúng ra thành một ngôn ngữ trung gian gọi là Microsoft Intermediate Language (MSIL).