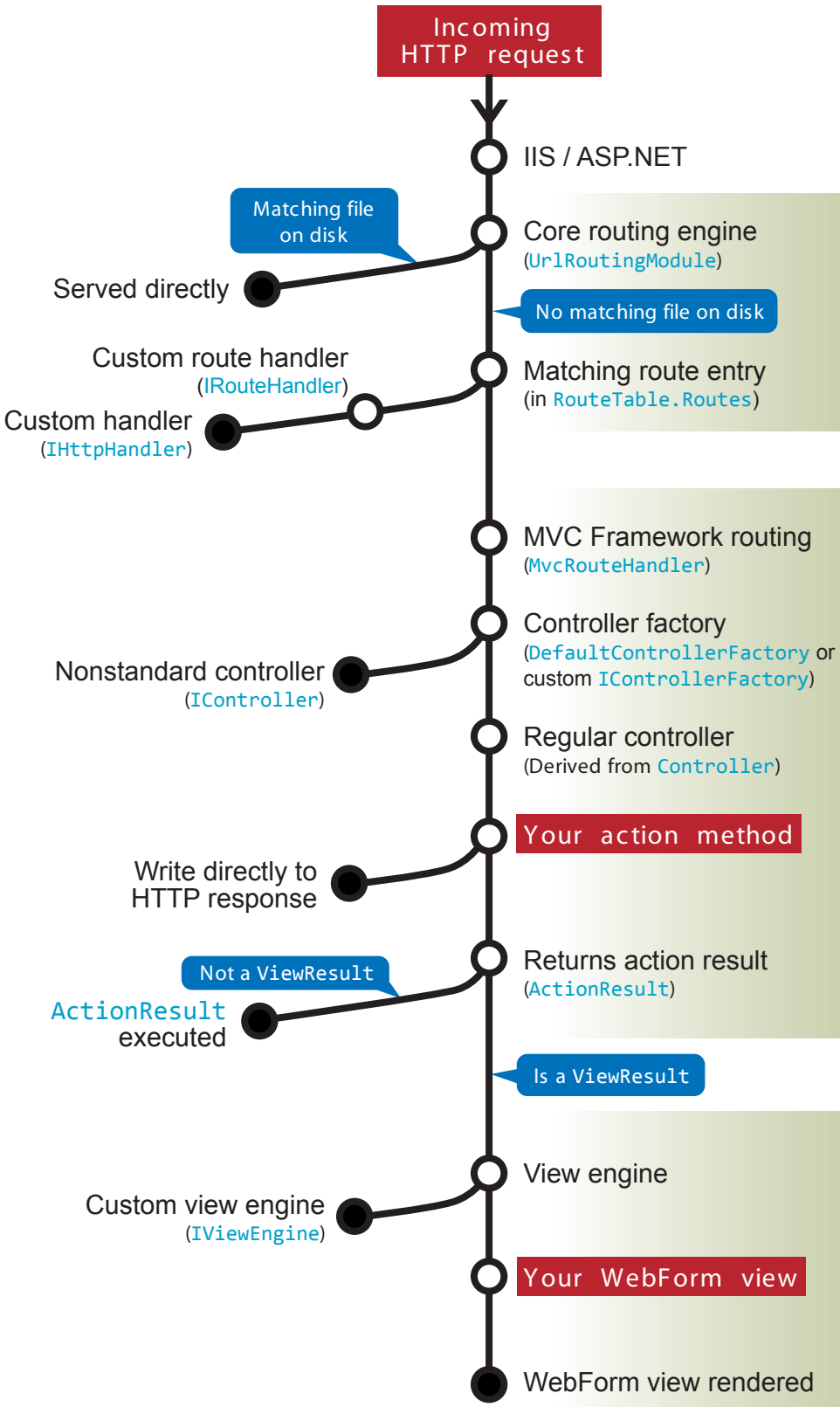
ASP.NET MVC: The Request-Handling Pipeline



1: ROUTING

PURPOSE:
Parses incoming URLs according to your configured routing schema, and then dispatches control to the matching route handler (typically, a controller)

BENEFIT:
Lets you define an arbitrary clean, human-readable, search-engine optimized (SEO) URL structure independently of the code that handles those URLs

2: CONTROLLERS and ACTIONS

PURPOSE:
Controllers are collections of "action methods", each of which is a .NET method that hosts your logic for handling a particular type of request (such as saving a record). An action method may call your model layer to load/save data and to invoke business operations, and then may produce output by rendering a view.

FEATURE:
Can use "filters" (not shown on diagram) to tag reusable behaviours before and after authorization, action methods, action results, and unhandled exceptions

FEATURE:
Can enforce access control by applying an authorization filter to an action method

FEATURE:
Automatically parses incoming HTTP data into .NET objects ("model binding")

BENEFIT:
Structured to make application logic simple and testable (e.g., for Test-Driven Development)

3: VIEWS

PURPOSE:
Render data supplied by the controller as HTML via a templating mechanism

FEATURE:
Use ASPX templates which support Visual Studio IntelliSense and master pages, or use your own custom view engine

FEATURE:
Keeps HTML construction simple and separate from other application concerns

BENEFIT:
No sophisticated server-side controls involved, so can be edited by HTML designers who need no knowledge of ASP.NET. No viewstate, no postbacks, and no ID mangling, so works great with jQuery or any other client-side toolkit.

redgate®

Original diagram taken from
Pro ASP.NET MVC Framework
(Steven Sanderson, Apress)