

Carimbo de data/hora	Pontuação	Name and Student ID	What is the name of the file you evaluated?	How would you rate your confidence that there is a DISPERSED COUPLING smell in the code?	Mention why (for instance, which code elements) you gave this rating.
27/04/2025 14:35:52	UFMG #32	Monitor.java		1	Não consegui identificar a presença de dispersed coupling. Parece que vai ser usada em vários lugares, mas como é relacionado a conexão, presumo que existe uma classe que controla as conexões criadas e monitoradas.
28/04/2025 22:40:23	UFMG #21	Monitor		2	The class calls several methods from other classes/API, but I am not sure if this is a problem here.
29/04/2025 12:27:46	UFMG #13	StatisticSlot		2	Apesar de possuir um alto acoplamento, o uso das classes importadas não me parece desordenado ou disperso.
29/04/2025 14:00:58	UFMG #13	ServiceLoader		2	Faz a chamada de muitos objetos para uma classe pequena. Porém, essas chamadas não me parecem descentralizadas.
29/04/2025 14:06:08	UFMG #32	HtmlTreeBuilderStateTest		1	Não importa muitos objetos externos, e todos eles estão bem concentrados em poucas tarefas.
29/04/2025 14:08:17	UFMG #13	AssertUtil		1	Os métodos que utilizam as classes externas me parecem bastante coesos.
				5	Código repetido, como em: return new ObjectConstructor<T>() { @Override public T construct() { return (T) new LinkedHashSet<Object>(); } }
01/05/2025 17:52:52	UFMG #6	ConstructorConstructor			Método get muito longo e com várias funcionalidades. Pode ser quebrado em mais de um método.
01/05/2025 18:21:41	UFMG #23	PageModelExtractor		3	Existem várias referências a outras classes, que vêm de outros módulos/pacotes e que interagem entre si.
02/05/2025 09:23:49	UFMG #30	Excluder		1	Não há uso de bibliotecas ou módulos externos ao Gson ou ao Java Core.
14/05/2025 17:40:51	UFMG #3	AssertUtil		2	There are 7 dependencies, but all of them are from the std of the language, it's a high number of dependencies but since they're from the std it's ok. All in now if the std of the language breaks you should probably reconsider the usage of the language as a whole.
14/05/2025 17:48:24	UFMG #11	ServiceLoader		5	O código já deixa carece de coesão, sua leitura é difícil, o que é característico de acoplamento disperso. Nesse sentido, a classe depende de muitos módulos do java como URL, InputStream, Collection, dentre outros, que não interagem muito entre si.
14/05/2025 17:50:17	UFMG #33	DBeaverCore		5	O código apresenta uma única classe com mais de 200 linhas, muitos métodos e muitas responsabilidades. Acredito ser um caso de Large Class. Além disso, apresenta um método chamado GetTempFolder com muitos IFs, caso de Long Method.
14/05/2025 17:51:57	UFMG #18	JsonAdapterAnnotationTypeAdapterFactory		2	O código é relativamente coeso, delega corretamente a criação de adaptadores conforme o tipo de instância (TypeAdapter, TypeAdapterFactory, JsonSerializer/JsonDeserializer), e aplica a opção nullSafe() de forma clara. As possíveis melhorias seriam pontuais (por exemplo, reduzir o uso de instanceof com um padrão mais orientado a registros ou lambdas), mas nada que constitua um cheiro forte ou prejuízo à manutenibilidade de forma significativa.
14/05/2025 17:53:17	UFMG #9	TreeTypeAdapter		5	A classe possui muitos atributos de outras classes. Atribui nota 4, pois: 1. O método load(Class<T>, ClassLoader) (~35 linhas) mistura três responsabilidades.
14/05/2025 17:53:54	UFMG #16	ServiceLoader		4	2. Temos Dois blocos catch (...) /* skip */ que suprimem erros, podendo esconder possíveis falhas 3. Esse eu não tenho certeza mas meu pensamento é que o set estático loadedUrls é mutável e acessado por múltiplas threads sem proteção, o que pode acarretar em um problema de concorrência
14/05/2025 17:57:38	UFMG #27	PreparedStatementHandler		4	O código apresenta somente uma classe que corta a comparação para a identificação de determinados code smells. Contudo, seus métodos todos dão override na classe inicial. Logo, é um caso de refused bequest.
14/05/2025 17:58:31	UFMG #2	AclException		2	A classe não apresenta de forma evidente o code smell. Desse modo, ela não compromete gravemente o princípio da coesão ou modularidade. No envio anterior não contelei exatamente este code smell portanto estou reenviando.
14/05/2025 17:59:59	UFMG #20	Stage		2	A classe Stage possui algumas dependências externas (como Runnable, Timer, Platform, entre outras), mas elas estão concentradas em torno de funcionalidades específicas (controle de execução e interface gráfica). O acoplamento existe, mas não é excessivamente disperso entre múltiplos módulos ou pacotes. A classe é coesa no contexto da lógica de fases/tarefas.
14/05/2025 18:01:15	UFMG #25	eval-llms-code-smells /ClientHandler		5	O código possui Dispersed Coupling, pois a classe tem 3 atributos que são instâncias de outras classes e em seu método run(), além de usar essas 3 instâncias, também instancia outros 3 elementos de outras classes além dessas.
14/05/2025 18:03:05	UFMG #29	EnablePluginCommand		2	Essa classe tem muitos imports, o que indica que ela depende de muitas outras classes. Entretanto os métodos não usam todos os imports na mesma, como é distribuído no código a refatoração não é tão difícil.
14/05/2025 18:04:12	UFMG #28	StatisticSlot		5	A classe StatisticSlot depende de diversos outros módulos, como em "context.getCurEntry().getOriginNode().increaseThreadNum()", e outras partes do código também, o que é um dispersed coupling. Além disso, esse código possui o code smell de Long Method, uma vez que o método "entry" é muito extenso e difícil de entender.
14/05/2025 18:04:20	UFMG #24	NioClientTest		2	É um código objetivo, mas comentários em mais de uma linguagem
14/05/2025 18:04:51	UFMG #22	Vault		5	Sim, pois a classe Vault apresenta muitas dependências
14/05/2025 18:05:38	UFMG #37	NioClientTest		2	Eu não acho que haja o code smell Dispersed Coupling no código, pois a classe NioClientTest não instancia nenhum método de outra classe em sua implementação. Não tenho certeza sobre se há um outro code smell na classe, mas acredito que não
14/05/2025 18:06:29	UFMG #5	NioClientTest		5	A classe NioClientTest só tem um método, main, que é um teste para um caso específico e, portanto, só tem objetos e métodos de outras classes. Não é o melhor jeito de implementar um teste.
					No envio anterior não contelei exatamente este code smell portanto estou reenviando.
14/05/2025 18:07:07	UFMG #20	CommandDecoder		3	A classe depende de várias outras, mas essas dependências estão relacionadas à lógica central de decodificação. Elas não estão espalhadas de forma desorganizada ou desconexas. Há algum acoplamento, mas ele é razoavelmente concentrado e coerente com a função da classe.
14/05/2025 18:07:22	UFMG #19	PlainPermissionManager		4	A classe "PlainPermissionManager" realmente depende de vários de módulos diferentes, como PlainAccessConfig e PlainAccessResource, entretanto, pode ser dito que ela faz de maneira centralizada, pois em teoria, ela maneja permissões para estes módulos, portanto, poderia-se argumentar pela ausência de Dispersed Coupling.
14/05/2025 18:07:50	UFMG #38	GeometryViewerRegistry		4	I think there is definitely Dispersed Coupling in this code, although I think the main problem is that the class is too long
14/05/2025 18:08:51	UFMG #41	HtmlTreeBuilderStateTest		3	Existe uma certa repetição, poderia criar uma função mais genérica com o corpo do html
14/05/2025 18:09:51	UFMG #17	PropertyElf		5	O método está dependendo de outras classes diferentes de forma "espalhada".
14/05/2025 18:10:04	UFMG #34	HazelcastPubSubStore		4	Minha confiança é 4. Esta classe interage com diversos componentes distintos para cumprir seu papel, como HazelcastInstance para publicação e subscrição, as interfaces PubSubListener e PubSubMessage para a troca de mensagens, e até PlatformDependent do Netty para criar um mapa concorrente. Essas interações com diferentes abstrações e bibliotecas, espalhadas pelos métodos, caracterizam o Dispersed Coupling, tornando a compreensão geral das suas dependências um pouco mais trabalhosa.
14/05/2025 18:10:35	UFMG #34	ServiceLoader		4	Minha confiança é 4. A classe ServiceLoader demonstra Dispersed Coupling ao interagir com múltiplas APIs de baixo nível do Java para carregar e instanciar serviços, como ClassLoader para obter recursos e carregar classes, URL, InputStream e BufferedReader para ler os arquivos de serviço. Adicionalmente, utiliza IOUtils para o fechamento de recursos, mostrando dependências espalhadas por diferentes funcionalidades do sistema para realizar sua tarefa principal.
14/05/2025 18:11:16	UFMG #26	PropertyElf		1	Acredito que não há Dispersed Coupling. Apesar de realizar chamadas de métodos de qualquer classe utilizada como parâmetro, a generalização funciona para todas as classes utilizadas.
14/05/2025 18:12:36	UFMG #20	HTTPSession		4	No meu outro envio não contelei exatamente este bad smell em específico, mas sim outros. Portanto estou reenviando.
14/05/2025 18:13:34	UFMG #16	DBeaverCore		5	A classe depende de diversos componentes externos. Essas dependências estão distribuídas por diferentes partes da lógica da classe, o que dificulta a compreensão global e a manutenção. A diversidade de módulos acoplados evidencia um acoplamento disperso.
14/05/2025 18:14:42	UFMG #12	AutoLocker		1	Dispersed Coupling - Não Encontrado
14/05/2025 18:17:06	UFMG #43	CodahaleHealthChecker		5	Long Method - O método "public File getTempFolder(DBRProgressMonitor monitor, String name)" possui muitas condicionais aninhadas Refused Bequest - Vários métodos sofreram override
14/05/2025 18:17:19	UFMG #32	PreparedStatementHandler		4	The class AutoLocker doesn't depend on multiple other modules or classes in a decentralized way. It seems to depend more specifically only on ScheduledExecutorService and vaultList.
14/05/2025 18:17:34	UFMG #17	CellFormulaTagHandler		3	Com certeza este código tem falhas no acoplamento, sendo muito amarrado com as funções que utiliza e sendo muito difícil de dar manutenção caso alguma mude.
14/05/2025 18:18:25	UFMG #27	PropertyElf		5	É possível ver que tanto os parâmetros do método registerHealthCheck quanto o corpo do método apresentar acoplamento com objetos especializados ao invés de interfaces por exemplo.
14/05/2025 18:20:07	UFMG #38	CodeCacheEventWalker		4	O código apresenta Dispersed Coupling porque a classe depende de várias outras classes dispersas em seus métodos, como ResultSetHandler, KeyGenerator, e Executor. Esse acoplamento espalhado torna a classe difícil de entender e de manter.
14/05/2025 18:21:38	UFMG #10	CodahaleHealthChecker		3	Não apresenta dispersed coupling.
14/05/2025 18:23:32	UFMG #41	DBeaverCore		5	Ele é uma classe utilitária que centraliza o acesso e manipulação de propriedades, evitando espalhar chamadas e dependências por várias classes. Isso significa que o acoplamento está concentrado, não disperso.
14/05/2025 18:25:21	UFMG #40	SingleRoomBroadcastOperations		4	The class directly accesses types from multiple unrelated modules: model, codocache, and logging. It creates, modifies, and interacts with Compilation, CodeCacheEvent, IMetaMember, and logging logic inline, without abstraction. This tight inter-class interaction across different concerns, especially within a single method (visit), confirms dispersed coupling.
14/05/2025 18:25:54	UFMG #25	ExcelWriteFillExecutor		5	No código, principalmente no primeiro método, o badsmell de Dispersed Coupling aparece diversas vezes. Sendo usados métodos de outras classes para fazer modificações em variáveis dentro da função.
14/05/2025 18:27:00	UFMG #9	HazelcastPubSubStore		2	Também existe o badsmell de missaging chains dentro desse mesmo método.
14/05/2025 18:28:20	UFMG #31	HtmlTreeBuilderStateTest		5	A classe DBeaverCore utiliza informações de 5 classes diferentes em sua implementação, indicando um smell do tipo Dispersed Coupling.
14/05/2025 18:29:34	UFMG #3	OmsPortalOrderServiceImpl		4	Há pouco contexto para definir a existência do code smell, mas é possível que ele ocorra por meio de SocketIOClient.
14/05/2025 18:30:05	UFMG #26	HTTPSession		2	Por mais que tenha muitos módulos na classe, elas parecem organizadas de maneira coesa, de maneira que cada método centraliza o uso de cada um dentro deles.
14/05/2025 18:30:23	UFMG #8	Monitor		5	Por mais que tenha muitos módulos na classe, elas parecem organizadas de maneira coesa, de maneira que cada método centraliza o uso de cada um dentro deles.
14/05/2025 18:30:38	UFMG #38	StatisticSlot		4	A classe analisada não depende de tantas classes externas assim, mas manipula e usa tipos e conceitos de ambas as bibliotecas de Pub/Sub. Pode ser um bad smell porque implica que alterações significativas em qualquer uma dessas dependências externas podem exigir modificações na classe em questão.
14/05/2025 18:31:14	UFMG #14	XxJobExecutor		2	A classe não parece possuir nenhum code smell óbvio
14/05/2025 18:33:02	UFMG #8	Code		5	A classe possui vários atributos de várias classes diferentes. Nos métodos são chamadas diversos métodos de diversas classes. Definitivamente tem um code smell de dispersed coupling.
14/05/2025 18:34:25	UFMG #26	CommandDecoder		5	Diversos métodos da classe utilizam outras chamadas de outras classes, gerando enorme dependência do funcionamento dessas e impossibilitando a manutenção do código.
14/05/2025 18:35:49	UFMG #42	HazelcastPubSubStore		1	O método não chama as funções que poderiam ser implementadas em sua declaração. Ele possui um método que tem sua funcionalidade. Contudo, o método caracteriza um long method
14/05/2025 18:35:54	UFMG #24	ExecutionSequencer		4	The class has dispersed coupling. It directly accesses and coordinates types from multiple distinct subsystems: context handling, resource wrapping, node statistics, slotchain callbacks, exception types, utility functions, and constants. These represent unrelated modules.
14/05/2025 18:36:27	UFMG #22	OmsPortalOrderServiceImpl.java		5	Esta classe está muito dependente de como os dados das classes job e admin são enviados, caso ocorra erro entre elas essa classe também será afetada.
14/05/2025 18:38:43	UFMG #19	CodeCacheEventWalker		2	Não parece haver o uso de muitos módulos diferente no código como um todo.
14/05/2025 18:39:03	UFMG #28	CodahaleHealthChecker		5	Os métodos da classe são altamente dependentes dos métodos de outras classes, portanto, acredito que haja o badsmell.
14/05/2025 18:40:06	UFMG #15	ResizeController		2	Tem pouco de acoplamento sim mas não me pareceu que foram usadas muitas classes nos métodos. Conteúdo no máximo 4 classes em cada método.
14/05/2025 18:40:46	UFMG #42	PageModelExtractor		5	Depende de vários métodos da Google
14/05/2025 18:41:11	UFOP #5	ExecutionSequencer		5	Várias chamadas e métodos grandes
14/05/2025 18:41:29	UFOP #16	Internet		1	A classe depende de fato de outros módulos, mas como ela é um visitor, não tem como escapar desse fato, sendo uma necessidade para que a classe cumpra sua função.
14/05/2025 18:41:39	UFOP #9	CodeCacheEventWalker		5	A classe implementada nesse código depende de muitos métodos que não são da própria classe, como "getHealthCheckProperties", "getMetricRegistry", "getProperty" e vários outros.
14/05/2025 18:42:06	UFOP #15	ResizeController		1	Não encontrado nenhum bad smell
14/05/2025 18:42:36	UFOP #42	PageModelExtractor		5	Tem muito code smell nessa classe. Tem dispersed coupling, tem métodos longos, tem long class...
14/05/2025 18:43:04	UFOP #21	Stage		4	A classe Stage parece depender fortemente de outros módulos para desenhar o estágio. Talvez isso seja necessário para que as demais classes não fiquem sobrecarregadas.
27/05/2025 15:30:39	UFOP #7	ExecutionSequencer		5	A classe ExecutionSequencer apresenta acoplamento disperso por depender intensamente de múltiplas outras classes e múltiplos níveis de chamadas indiretas
04/06/2025 19:42:11	UFOP #5	Throwables.java		4	Excesso de Overrides, pelo menos pra mim, 6 métodos de outras classes não seria muito viável
04/06/2025 19:44:10	UFOP #16	Internet		5	Acredito que o código possua o code smell large class. O código da classe é bem extenso e possui diversos métodos que poderiam estar em outras classes.
04/06/2025 19:46:38	UFOP #9	CodeCacheEventWalker		2	Apesar de ter vários "import" neste código, o que pode causar confusão se verificar muito rápido, não parece ter o code smell Dispersed Coupling, por não ter a dependência de muitas outras classes diferentes de maneira dispersa ou descentralizada.

Carimbo de data/hora	Pontuação	Name and Student ID	What is the name of the file you evaluated?	How would you rate your confidence that there is a DISPERSED COUPLING smell in the code?	Mention why (for instance, which code elements) you gave this rating.
04/06/2025 19:46:38	UFOP #2	EnablePluginCommand		4	<p>Existe dispersed coupling pois vários objetos criados na classe são na verdade importados de outra, além disso há um uso frequente de métodos que também são de outras classes.</p> <p>A maioria dos métodos depende de módulos de outras classes como as do trecho abaixo:</p> <pre>BarcodeMetadata getBarcodeMetadata() { Codeword[] codewords = getCodewords(); BarcodeValue barcodeColumnCount = new BarcodeValue(); BarcodeValue barcodeRowCountUpperPart = new BarcodeValue(); BarcodeValue barcodeRowCountLowerPart = new BarcodeValue(); BarcodeValue barcodeECLlevel = new BarcodeValue();</pre>
04/06/2025 19:53:39	UFOP #29	DetectionResultRowIndicatorColumn		4	<p>Apresenta um pouco de dispersed coupling quando percebemos que todos os métodos utilizam de uma ou mais classes para o funcionamento, mas nada alarmante, além de que a estrutura está bem organizada.</p>
04/06/2025 19:55:06	UFOP #7	SaSecureUtil		2	
04/06/2025 19:55:34	UFOP #17	PageModelExtractor.java		5	A classe PageModelExtractor tem acoplamento com vários outros serviços
04/06/2025 19:55:56	UFOP #19	EnablePluginCommand		4	Notei um grande número de imports no começo do arquivo. Além disso, houve um número considerável de chamadas dentro dos métodos.
04/06/2025 19:56:36	UFOP #18	Location.java		4	Classes muito grande e mesmo trecho de código aparece mais de uma vez em AccessLocation e Variable AccessLocation.
04/06/2025 19:57:09	UFOP #25	PlainPermissionManager		2	Olhando a classe PlainPermissionManager ela não aparenta ter acoplamento disperso por outros serviços.
04/06/2025 19:59:57	UFOP #9	SaSecureUtil		2	No primeiro contato fazendo a análise, eu fiquei um pouco confuso, mas verificando com calma, não parece ter o code smell Dispersed Coupling, apesar de ser um código grande com muitos tratamentos, eu diria que não tem o code smell analisado.
04/06/2025 20:00:19	UFOP #7	PlainPermissionManager		4	bastante dispersed coupling, em vários métodos como no PlainAccessResource tudo gira em torno de métodos de outras classes.
04/06/2025 20:00:53	UFOP #16	Internet		5	Há uma grande dependência de outros módulos durante o código. Vários métodos avulsos estão em uma única classe sem necessidade. Como por exemplo os método image e avatar.
04/06/2025 20:03:05	UFOP #7	ExcelWriteFillExecutor		4	bastante dispersed coupling, classe é muito dependente de biblioteca e classes externas. Um exemplo é o método doFill. Este método depende de várias classes e pacotes externos, como WriteContext, WriteSheetHolder, ExcelContentProperty, CellData, e WriteHandlerUtils.
					Acredito que exista uma grande dependência dos métodos importados das classes abaixo:
					@Resource private XxlJobGroupDao xxlJobGroupDao; @Resource public XxlJobInfoDao xxlJobInfoDao; @Resource public XxlJobLogDao xxlJobLogDao;
					A maioria dos métodos da classes necessita de métodos das classes acima o que pode gerar dependências múltiplas.
04/06/2025 20:04:22	UFOP #30	Stage		5	A classe Stage possui muitas dependências.
04/06/2025 20:04:47	UFOP #22	RocketMQMessageHandler.java		3	Acredito que não há, porém com dúvidas. Há uma classe que depende de múltiplas outras classes, entretanto estas possuem tratamento de erros, com blocos try/catch e lançamentos de exceções
04/06/2025 20:09:05	UFOP #12	RocketMQMessageHandler		1	Acredito que não haja um code smell, pois por mais que tenhamos algumas classes sendo instanciadas não é um número exagerado
04/06/2025 20:13:59	UFOP #1	DetectionResultRowIndicatorColumn		4	A classe DetectionResultRowIndicatorColumn se enquadra na definição de Dispersed Coupling, possuindo muitos e variados módulos em todo o escopo da classe, o que causa uma descentralização no código e resulta em falta de coesão.
04/06/2025 20:14:48	UFOP #30	TransactionMQProducer		5	A classe TransactionMQProducer possui muitos métodos sobrescritos (@Override) ou ignorados (@Deprecated).
04/06/2025 20:15:07	UFOP #23	MapTypeAdapterFactory		5	A característica de chamar vários métodos de outras classes mas a comunicação não é tão firme.
04/06/2025 20:16:20	UFOP #17	Internet.java		5	A classe depende fortemente de outros módulos e classes
04/06/2025 20:21:15	UFOP #15	Vault		5	Dependências Múltiplas é o principal code smell desse código.
04/06/2025 20:25:46	UFOP #4	Location		5	Atribui a nota 5, pois de fato o código está bem descentralizado, tornando até um pouco difícil de entender.
04/06/2025 20:26:34	UFOP #33	ExcelWriteFillExecutor.java		3	Não função dealAnalysisCell utiliza pelo menos de 3 módulos diferentes. Porém olhando o contexto do método e da aplicação analisada, pode ser que não seja um code smell e seja uma solução aplicável neste ponto.
04/06/2025 20:26:59	UFOP #10	Throwables		1	O código com seus métodos e suas classes não apresentam ter múltiplas dependências de outros métodos ou classes, o que indicaria a não presença de um Dispersed Coupling
					a classe JobLogControl tem muita responsabilidades: gerenciamento de logs
					Filtragem por permissões
04/06/2025 20:31:44	UFOP #14	eval-lms-code-smells /JobLogController.java		5	Comunicação com executores
					Paginação e filtros complexos
					Lógica de limpeza de logs
04/06/2025 20:33:19	UFOP #4	MapTypeAdapterFactory		2	Atribui a nota 3, pois não consegui identificar múltiplas dependências ou descentralização.
04/06/2025 20:33:21	UFOP #28	SaSecureUtil		1	O método não possui Dispersed Coupling pois não faz chamada a métodos de classes tão diferentes.
04/06/2025 20:35:44	UFOP #28	JobLogController		4	O método possui o Dispersed Coupling pois faz chamadas a métodos de muitas outras classes.
04/06/2025 20:37:07	UFOP #26	Throwables		1	Não é um caso de Dispersed Coupling, a classe aparenta não depender de outras classes ou métodos.
04/06/2025 20:47:00	UFOP #3	Location		5	Muito código repetido entre as várias classes internas (especialmente na criação dos StackSaver. Além disso, o arquivo é grande por causa de tantas classes aninhadas. Isso tudo pode dificultar a manutenção e o entendimento do código.